Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Facultad de Ingeniería de Sistemas e Informática

Escuela Académico Profesional de Ingeniería de Sistemas

Herramienta que automatiza la configuración de conectores para la integración de sistemas heredados

TESINA

Para optar el Título Profesional de Ingeniero de Sistemas

AUTORES

Henry William FLORES ESPINOZA

Héctor Andre MANRIQUE PULACHE

ASESOR

Erwin MAC DOWALL REYNOSO

Lima, Perú

2008

Referencia bibliográfica

Flores, H. & Manrique, H. (2008). *Herramienta que automatiza la configuración de conectores para la integración de sistemas heredados*. Tesina para optar el título profesional de Ingeniero de Sistemas. Escuela Académico Profesional de Ingeniería de Sistemas, Facultad de Ingeniería de Sistemas e Informática, Universidad Nacional Mayor de San Marcos, Lima, Perú.

Este trabajo esta dedicado a nuestros familiares por su gran apoyo que siempre nos han brindado.

AGRADECIMIENTOS

En primer lugar queremos agradecer a nuestro asesor el Lic. Erwin Mac Dowall Reynoso, por su dedicación y orientación para poder cumplir los objetivos trazados en el presente trabajo.

También queremos agradecer a nuestros compañeros y docentes de la UNMSM, por compartir sus conocimientos durante todo el transcurso de nuestra carrera universitaria.

Por ultimo queremos agradecer muy especialmente a nuestros padres por ese apoyo incondicional que siempre nos han brindando, a nuestros hermanos y a todas aquellas personas que de alguna forma aportaron sus conocimientos para poder llegar a cabo nuestro trabajo.

Herramienta que automatiza la configuración de conectores para la integración de Sistemas Heredados

RESUMEN

A efectos de poder adaptar los procesos de negocio a nuevos escenarios tecnológicos, las organizaciones se han visto en la necesidad de integrar sus sistemas, con la finalidad de poder generar mayores ventajas competitivas. Sin embargo, esta es una tarea ardua debido a que muchos de los sistemas están basados en arquitecturas monolíticas y rígidas. Este es el caso de los Sistemas Heredados, los cuales pese a que son considerados obsoletos, son de misión crítica para la organización. Otras de las características de estos tipos de sistemas, es que las organizaciones se resisten a reemplazarlos, debido a que, a lo largo del tiempo han demostrado su fiabilidad y seguridad.

Debido a las características descritas, es importante para las organizaciones mantener la lógica de negocio que se encuentra incluida dentro de estos tipos de sistemas, ya que es muy difícil en la mayoría de casos tratar de migrarlos, pues no siempre cuentan con la documentación completa que se pueda utilizar para la construcción de un nuevo sistema.

El objetivo de nuestro trabajo es crear una herramienta que permita la configuración de conectores para la integración de Sistemas Heredados, utilizando para ello adaptadores, que permitan la transformación de mensajes entre distintos sistemas heterogéneos, y Servicios Web para llevar a cabo la interoperabilidad.

Uno de los beneficios de aplicar este tipo de integración, es que no hay que realizar cambios en los sistemas para poder comunicarlos unos con otros, sino que a través de conectores, se adaptan interfaces estándares que hacen posible la integración de los sistemas.

Palabras Claves: Sistemas Heredados, Wrappers, Servicios Web, Conectores.

Tool that automates the configuration of connectors for the integration of Legacy Systems

ABSTRACT

To effects of being able to adapt the processes of business to new technological scenes, the organizations have seen the necessity to integrate their systems, in order to be able to generate greater competitive advantages. However, this is a difficult task because many of the systems are based on monolithic and rigid architectures .This is the case of the Legacy Systems. Which although they are considered obsolete, are of mission critical for the organization. Other of the characteristics of these types of systems, It is that the organizations refuse to replace them, due to the fact that, throughout the time they have demonstrated their reliability and safety.

Due to the described characteristics, it is important for the organizations to maintain the logic of business that is included inside these types of systems, because in the majority of cases it is very difficult try to migrate them because they don't always have the complete documentation that you can use for the construction of a new system.

The aim of our work is to create a tool that makes possible the configuration of connectors for the integration of Legacy Systems, using adapters, which allow the transformation of messages between different heterogeneous systems and Services Web to take the interoperability

One of the benefits of applying this kind of integration is that it is not necessary to make any changes to the structure of the systems to communicate one with the others, but through the connectors, they adapt standard interfaces that make possible the integration of the systems.

Key words: Legacy Systems, Wrappers, Web Services, Connectors.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

Capítulo 1: Introducción

1.1 Antecedentes

La comunicación entre sistemas, surge de la necesidad de: compartir datos, integrar procesos, disminuir los costos de administrar sistemas por separado, aumentar el nivel de automatización de los procesos del negocio, generar entornos de trabajo colaborativos e integrar plataformas tecnológicas.

Sin embargo existen varios aspectos que considerar al integrar sistemas. Uno de ellos es la estructura de los sistemas, es decir existen sistemas no estructurados que combinan la capa de presentación con la lógica de negocio y el acceso a los datos, este es el caso de los sistemas heredados.

Estos sistemas presentan un esquema vertical, en el que cada uno opera independientemente, sin la participación de unos con otros. Así mismo, otra de las características de estos sistemas es que son antiguos, operan sobre plataformas obsoletas y no cuentan con la documentación actual del sistema.

Pese a que el mantenimiento y seguimiento de estos sistemas son costosos, son parte vital del negocio y en muchos de los casos se prefiere evitar su reemplazo por un sistema más moderno.

Existen diversas estrategias de integración de sistemas heredados, como son: redesarrollo de los sistemas, redesarrollo por módulos, integración mediante la técnica del Wrapping e integración utilizando Servicios Web. Sin embargo, la implementación depende en gran medida del ámbito al cual se va aplicar, se debe evaluar el impacto del cambio, el costo ha invertir, y dependiendo de estos factores considerar cual de las integraciones es las mas adecuada, para poder llevar a cabo su implantación.

1.2 Definición del Problema

El principal problema que presentan los Sistemas Heredados es su naturaleza heterogénea, es decir muchos de estos sistemas han sido desarrollados por diferentes proveedores, los cuales han desarrollado los aplicativos en diferentes plataformas

de hardware y software, esto por consiguiente, genera un conflicto al intentar comunicar un sistema con otro. Pese a que estos sistemas pueden ser reemplazados por sistemas basados en tecnologías mas modernas, no es llevado a cabo en la mayoría de los casos, esto debido a la dificultad de migrar tipos de datos no estructurados. Esto debido a que los sistemas heredados poseen una naturaleza monolítica, en la cual se mezcla, la lógica de negocios con el acceso a la base de datos y la presentación, todo esto en una misma capa de desarrollo.

Otro de los inconvenientes para migrar sistemas heredados es que estos sistemas son antiguos operan bajo plataformas obsoletas, relativamente primitivas.

Es por estos motivos que en la mayoría de los casos las organizaciones prefieren evitar cualquier modificación que pueda poner en peligro el funcionamiento del Sistema.

1.3 Objetivos

1.3.1 Objetivo Principal

Creación de una herramienta que automatiza la configuración de conectores para la integración de Sistemas Heredados, con la finalidad de intercomunicar distintos sistemas, independientemente de la plataforma en la que se encuentren operando.

1.3.2 Objetivos Secundarios

- Creación de un Wrapper, que se encargara de mapear la información recibida, para posteriormente convertirla a un formato entendible para el servicio deseado.
- Estudiar y evaluar las diferentes formas de integrar Sistemas Heredados.

1.4 Justificación

Un problema muy común que se suscita en el ámbito informático, es la cantidad de información disponible en diferentes fuentes de datos heterogéneas, en muchos de los casos estas fuentes se encuentran divididas en diferentes plataformas

Hardware y Software, es por esta diversificación que la tarea de integrar sistemas se torna compleja y difícil.

Estos sistemas que atienden y responden correctamente a las necesidades del negocio, por lo general son ejecutados y operan sobre plataformas obsoletas. Cualquier mantenimiento a estos tipos de sistemas, genera grandes costos a la organización, tanto para la asignación del personal, como también en el caso de querer reemplazar el sistema por una más moderno. Sin embargo, estos sistemas son de vital importancia para el correcto funcionamiento de los procesos del negocio, llegando en la mayoría de los casos a crearse una dependencia mutua entre ambos.

1.5 Propuesta

La propuesta de trabajo que se presenta en esta tesis, es la de desarrollar una herramienta que automatice la generación de conectores para la integración de Sistemas Heredados, con la finalidad de intercomunicar distintas aplicaciones independientemente de la plataforma utilizada.

Se construirá un Wrapper para la conexión entre el Cliente Web y el servicio deseado. El Wrapper se encargara de realizar el mapeado de la información e interpretara los datos de entrada para ajustarlos a un formato basado en estándares XML.

Del lado del cliente, se generara un Cliente de Servicio Web, el cual podrá comunicarse con el Wrapper, para que esto sea posible, el Wrapper será publicado como Servicio Web. Para la comunicación se utilizaran mensajes SOAP, los cuales ubicaran los servicios deseados en el repositorio de Servicios.

El Wrapper mapeará la información para acceder a los servicios (COM+), sin embargo, estos componentes no necesariamente se encontraran en el mismo servidor en que se encuentra ubicado el Wrapper, para poder acceder a ellos se utilizaran llamadas a procedimientos remotos mediante Proxys, de esta forma el Wrapper podrá realizar las invocaciones a los métodos deseados como si fueran locales, es decir como si los componentes se encontraran alojados en el mismo servidor que se encuentra el Wrapper.

En el Wrapper se configuraran los servicios que se llamaran desde el Cliente

Web, así como también los parámetros de entrada y de salida de cada transacción a realizar.

Con el mapeador de información se lograra crear patrones de unificación de formatos, al recibir los datos entrantes estos serán convertidos en estructuras únicas para poder ser tratadas y enviadas al servicio deseado, para luego después de obtener la respuesta volver a interpretarla y devolverá a la aplicación solicitante en el formato que maneja.

Una de las ventajas de aplicar este tipo de integración es que los Sistemas Heredados no sufren modificaciones, al contrario, a través de aplicativos intermediarios se transforma la información a un formato que sea compresible por otros sistemas.

El alcance de nuestra propuesta de trabajo contempla la validación de los siguientes puntos:

- Creación de Wrapper que permita la integración de los Sistemas.
- Creación del Cliente de Servicio Web, del lado del Cliente, para la comunicación con el Wrapper, que es publicado como Servicio Web.

1.6 Organización de documento

La organización general de la presente tesina esta estructurada por un conjunto de 6 capítulos, descritos brevemente a continuación:

En el **Capítulo 2**, presentamos un estudio sobres las diferentes estrategias de integración de sistemas heredados, y la problemática existente al tratar de comunicar sistemas de distintas tecnologías.

En el **Capítulo 3**, detallamos el método para utilizar conectores que hacen posible la integración entre sistemas.

En el **Capítulo 4**, detallamos nuestra propuesta de trabajo, abordamos los conceptos principales de la arquitectura conceptual y las herramientas que se utilizaran para su desarrollo.

En el **Capítulo 5,** abordamos la implementación de nuestra propuesta de trabajo, se detalla las estructuras y herramientas empleadas para la integración de los sistemas a comunicar.

En el **Capítulo 6,** se presentan las conclusiones, recomendaciones y trabajos futuros de la presente tesis.

Capítulo 2: Marco Teórico

En este capitulo detallaremos los conceptos que sirvieron de base para la sustentación de nuestro trabajo, se mencionaran y explicaran las estrategias de integración de sistemas heterogéneos y heredados. Así mismo, se detallara las tecnologías relacionadas a la técnica del Wrapping, y también se detallara el funcionamiento de los Servicios Web.

2.1 Integración de sistemas

Existen algunos inconvenientes para poder realizar la comunicación entre sistemas de información, esto debido a que los sistemas presentan, en la mayoría de los casos, las siguientes características: Autonomía, heterogeneidad y la presencia de Sistemas Heredados.

La autonomía se presenta cuando, los sistemas operan de forma independiente, sin tomar en consideración la integración con otros sistemas. Esto por supuesto trae como consecuencia el crecimiento de la heterogeneidad, debido a que como operan sin comunicación con otros sistemas, se tiende a manejar herramientas y plataformas de desarrollo distintas para cada sistema.

La Heterogeneidad se presenta cuando se desarrollan sistemas con características distintas entre si, ya sea en componentes Hardware o Software, [AM 2002].

Se puede presentar heterogeneidad en:

Heterogeneidad Tecnológica: como lo son las diferentes herramientas Software y Hardware utilizadas, diferentes protocolos de comunicación, plataformas de desarrollo.

Heterogeneidad Sintáctica: diferencias e incompatibilidades en la representación de los datos, diferencias en las instrucciones utilizadas.

Heterogeneidad Semántica: diferencias en las representaciones de dominio.

Integrar aplicaciones en ambientes heterogéneos (lenguajes, hardware, sistemas operativos, bases de datos, archivos convencionales) para que interactúen

en forma fluida y se adapten conforme evolucionan o cambian los sistemas participantes o los procesos de negocios, requiere de un conjunto de instrumentos y procedimientos tanto para el desarrollo del sistema, como para la administración y el mantenimiento durante de su ciclo de vida. Para ello existen estrategias de integración que hacen posible la intercomunicación entre sistemas, por ejemplo se pueden utilizar interfaces de comunicación punto a punto o Middleware orientados a mensaje.

Estas estrategias de integración tienen como principal objetivo facilitar el desarrollo de sistemas a gran escala, los cuales se encuentran integrados por distintos componentes y soportados por entornos diferentes.

Anteriormente se menciono dentro de las características de los sistemas, la presencia de los Sistemas Heredados, en [AM 2002] definen a un sistema heredado, a aquel sistema, cuyo funcionamiento es esencial para que una empresa pueda operar normalmente en las actividades que dicho sistema atiende, es decir, son de misión crítica para la organización.

Los sistemas heredados constituyen un activo para la organización, por ello mismo como todo activo debe ser objetos de cuidados, mantenimientos y adecuaciones a los constantes cambios que sufren los negocios, sin embargo una de las características de estos sistemas es que son antiguos, por ello mismo son considerados obsoletos y hasta podría decirse primitivos.

Estos sistemas fueron desarrollados con el objetivo específico de solucionar los requerimientos funcionales dejando de lado aspectos importantes como lo son la evolución, portabilidad y compatibilidad con otros sistemas.

A continuación se mencionan las características de los sistemas heredados:

- Son sistemas desarrollados con lenguajes de programación de muchos años atrás.
- Estos sistemas son de tipo monolíticos, no están desarrollados bajo una arquitectura basada en capas, muy al contrario su arquitectura mezcla las funciones de acceso a los datos con la lógica de negocios y capa de presentación a los usuarios, en un mismo nivel.
- Cada uno de estos sistemas operan en forma independiente, por separado, uno

de otros, presentado así una integración normalmente vertical.

- Son sistemas por lo general de misión crítica para la organización.
- Son sistemas complejos y grandes, que incluyen dentro de su codificación reglas del negocio y los procedimientos operativos para llevar a cabo las tareas solicitadas.
- Presentan escasa o nula documentación, que haga referencia a las funcionalidades del sistema.

Es por estas características, que presentan estos sistemas, que las organizaciones prefieren continuar operando con ellos a pensar en optar por unos nuevos basados en tecnologías más modernas.

En la Figura 2.1 se muestra las diferentes clases de arquitectura, para el desarrollo de sistemas:

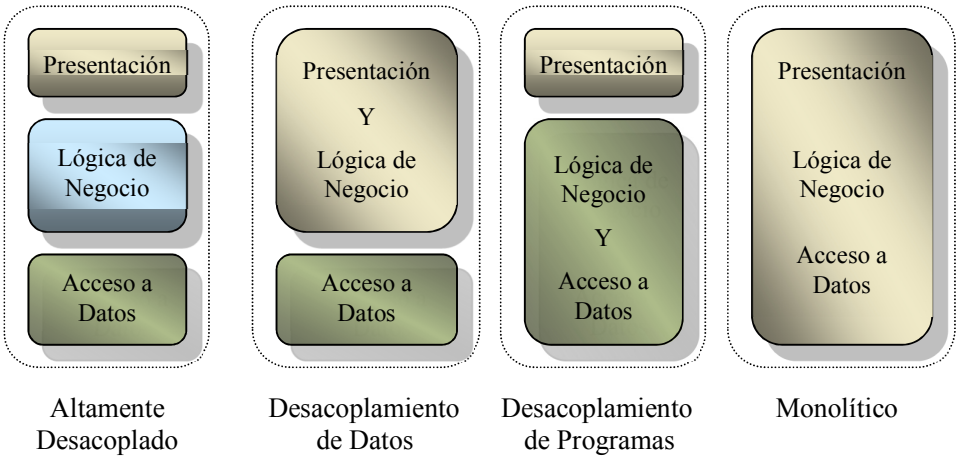


Figura 2.1: Clasificación arquitectura de sistemas.

Fuente: [AM 2002].

Altamente Desacoplados, esta clasificación es conocida como la arquitectura de 3-Capas, en donde se encuentran separados la lógica de presentación, lógica de negocio y acceso a datos. Este tipo de arquitectura presenta una mayor facilidad para el mantenimiento del sistema.

Desacoplamiento de datos, considerado semi-estructurados o arquitectura de 2-Capas, en este tipo de sistemas es posible acceder directamente a los datos.

Desacoplamiento de programas, también considerado semi-estructurado o

de 2-Capas.

Monolítico, sistema no estructurado, en este tipo de arquitectura se mezclan todos los niveles, la presentación, lógica y acceso a datos, lo que genera mayores problemas para dar mantenimientos al sistema.

A continuación se detalla los inconvenientes o riesgos al intentar migrar Sistemas Heredados:

- Altos costos de desarrollo: por lo general los Sistemas Heredados, son sistemas con un tiempo de vida muy largo, fueron desarrollados inicialmente bajo plataformas o lenguajes de programación relativamente primitivos, motivo por el cual es difícil encontrar personal que tenga conocimiento de las herramientas utilizadas para el desarrollo de estos sistemas.
- Grandes inversiones de tiempo para poder adaptarlos a las necesidades de la organización.
- Son dependientes de los fabricantes.
- Falta de documentación: debido a la naturaleza cambiante del mercado, los negocios han tenido que realizar modificaciones en sus sistemas para mantenerse en competitividad, estos cambios han hecho perder el estado inicial con el fueron entregados estos sistemas, y si existieran especificaciones, es mas probable que estas no cuenten los detalles desde su fase inicial, es decir desde que fueron puestos en producción.
- Debido a que estos sistemas fueron adecuándose a los requerimientos del negocio, se ha vuelto más fuerte la relación sistema – proceso de negocio.

2.2 Estrategias de Integración de Sistemas Heterogéneos

2.2.1 Un único proveedor

Esta estrategia consiste en diseñar un sistema o parte de un sistema utilizando un único proveedor para su desarrollo. Con esta estrategia se logra reducir la heterogeneidad tecnológica, sintáctica y semántica.

La principal desventaja de utilizar este tipo de sistemas es que casi siempre los proveedores utilizan tecnologías y procedimientos que están relacionados a su

plataforma, esto trae como consecuencia que el mantenimiento y actualización del sistema sea muy complicado.

Otro problema que surge al utilizar esta estrategia es que los proveedores difícilmente están especializados y preparados para resolver todas las necesidades de la organización.

2.2.2 Creación de Bases de Datos Centralizadas

Consiste en unificar o centralizar, todas las bases de datos que utilizan los diferentes sistemas heterogéneos dentro de una única base centralizada, esto reduce la duplicidad que puede existir en la información, sin embargo surgen problemas por la existencia de modelos de datos propietarios y problemas de acceso a la Base de Datos, debido a que se incrementa el número de aplicaciones con acceso a las Bases de Datos, ver Figura 2.2.

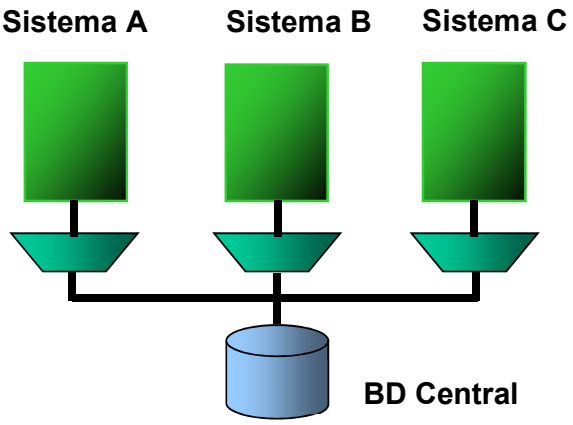


Figura 2.2: Bases de datos centralizadas.

Fuente: Creación Propia.

2.2.3 Interfaces punto a punto

Consiste en la construcción de interfaces por cada par de sistemas que se quieren comunicar, el esquema de esta estrategia de integración lógicamente es una red donde cada nodo representa una aplicación que necesita intercambiar información con otro nodo y con cualquiera de los demás nodos y para ello necesita generar una interfaz para la interconexión con cada nodo.

Esto permite que un sistema pueda invocar funciones de otro sistema, como

si esas funciones fuesen parte del sistema que ejecuta la invocación. Estas interfaces tienen o manejan varias formas de implementación que pueden ser conectores, adaptadores o Gateways, y son independientes de los esquemas tecnológicos y de la forma en que estén implementadas las demás conexiones entre los nodos.

Sin embargo una de las grandes desventajas de esta solución es que resulta realmente cara ya que el número de interfaces crece exponencialmente a medida que se quieran incrementar más nodos (aplicaciones) a la red. Si se quisieran interconectar "n" aplicaciones se requeriría realizar n*(n-1) interfaces de conexión.

Por ello el mantenimiento en este tipo de integración es realmente un problema, ya que la modificación en un nodo implica el cambio en los demás nodos que interactúan con este, ver Figura 2.3.

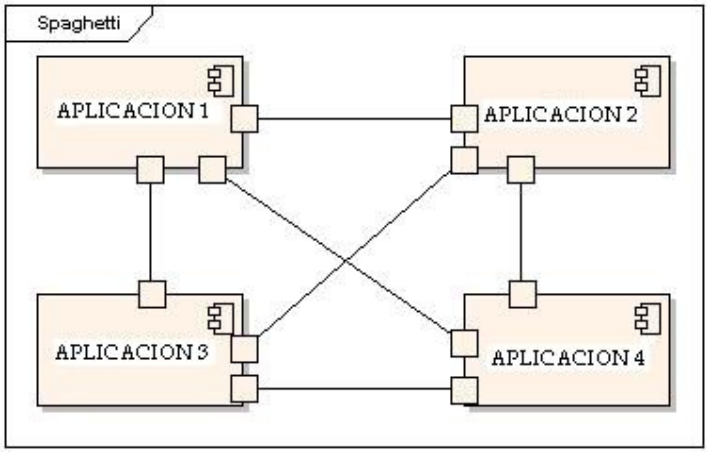


Figura 2.3: Modelo Spaghetti, integración tipo red.

Fuente: Creación Propia.

2.2.4 Hub and Spoke

Con el surgimiento de los modelos tipo "Hub", comienzan a proliferar el nacimiento de los "Middlewares". En este modelo, cada aplicación se conecta a un Hub, en el que residen todos los elementos de integración requeridos: "Brokers", soporte de mensajes, el repositorio de las operaciones o servicios disponibles, repositorio de macro operaciones, el motor de orquestación y el soporte de conectores y manejo transaccional, como los servicios más importantes.

Esta solución requiere interfacear cada aplicación con el concentrador de

mensajes a través de un adaptador Spoke. En este modelo cada aplicación solo tiene una interfaz programada con el "Hub" o con el concentrador de mensajes.

Las aplicaciones se comunican publicando un mensaje en el "Hub" y este publica esos mensajes para que sean utilizados por quienes lo necesiten. Los mensajes que se almacenan en la cola solo son recibidos por los que necesitan obtenerlos independientemente de cuantos mensajes se intercambien en ese momento, ver Figura 2.4.

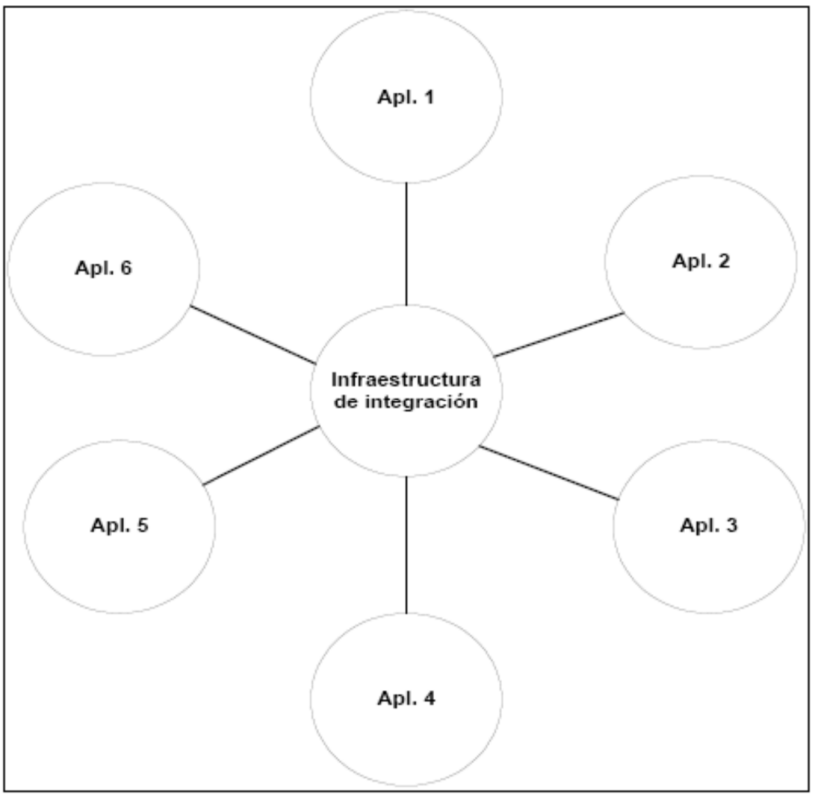


Figura 2.4: Modelo de integración tipo Hub (concentrador de mensajes).

Fuente: Creación Propia.

En muchos casos los productos que ofrecen tecnología de Middleware pueden también proveer servicios como transacción de mensajes y transformación de mensajes (uso de Brokers).

A los efectos de este trabajo tienen particular interés las tecnologías para la integración de aplicaciones. En forma general dichas tecnologías se pueden agrupar en: Enterprise Application Integration (EAI), se trata de productos y "Frameworks" para facilitar la integración de aplicaciones; y los que englobaremos bajo el nombre

de “Web Services”, que son los instrumentos que permiten la interoperabilidad a nivel de Web. Si bien este estudio las presenta separadas en grupos diferentes, en la práctica los productos que se aplican para EAI extienden sus funcionalidades para soportar “Web Services” y con ellos interoperabilidad a nivel de Internet.

2.2.5 Middlewares Orientados a Mensajes (MOM)

Los Middlewares orientados a mensajes pueden ser vistos como una extensión natural del paradigma del paquete de comunicaciones que prevalece en las capas inferiores de la red en el modelo OSI.

A diferencia de RPC y de la orientación a objetos los Middleware orientados a mensajes (MOM), son una forma de comunicación asíncrona, en el cual el remitente que envía el mensaje, no se bloquea esperando recibir una respuesta del receptor. Si el mensaje servicio ofrece persistencia y confiabilidad entonces el receptor no necesita estar en línea cuando el mensaje es enviado. Los mensajes que se intercambian generalmente no especifican o no pertenecen a un tipo de dato determinado ya que la estructura interna del mensaje es responsabilidad de la aplicación.

En los MOM tradicionales, los mensajes son dirigidos a los destinatarios respectivos, aunque el remitente o el receptor no estén conectados en ese momento, es decir, no es necesario que estén sincronizados para comunicarse. Esto puede ser menos adecuado en un área amplia, en gran escala de sistemas y puede ser ventajosa para desligar fuentes con respecto a los nombres, para que puedan ser mutuamente anónimos uno al otro, ver Figura 2.5.

Una típica manera de diseñar este tipo de sistemas, donde se publican y suscriben mensajes, consiste en publicar los mensajes en toda la red y solo los interesados se suscriben según sea su necesidad; entonces la red solo redirige los mensajes a los que se han suscrito a dicho recursos. Esto requiere que el servicio de transporte del mensaje comprenda los mensajes internos, aunque algunos sistemas manejan a nivel del mensaje una línea de asunto, entonces cuando el sistema lee esta línea puede ignorar el mensaje en caso sea necesario.

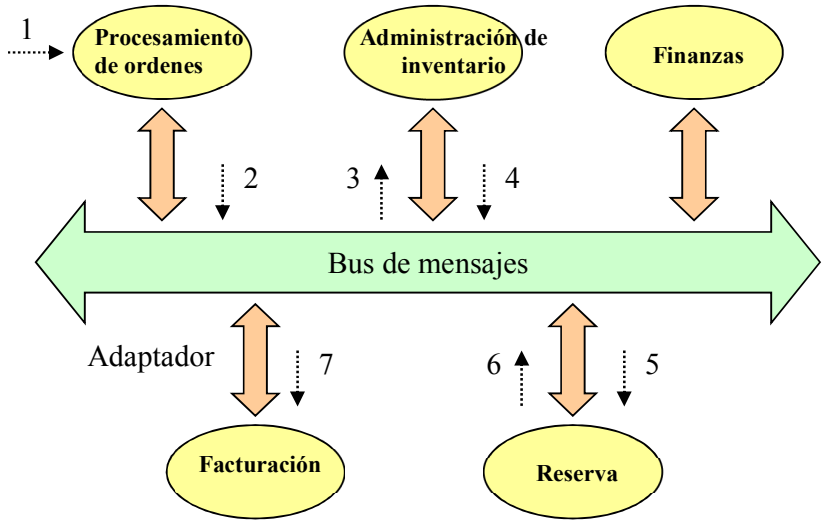


Figura 2.5: Middleware Orientado a Mensajes.

Fuente: [HM 1996].

2.2.6 Broker de Mensajes (Message Broker)

Esta estrategia es una evolución de la estrategia anterior, en la cual un controlador coordina los movimientos de información entre los sistemas integrados acorde con el flujo de procesamiento definido para los sistemas. Este controlador explícitamente coordina el flujo de comunicación entre sistemas que se interrelacionan. Un Message Broker es un ejemplo típico de un controlador o intermediador.

Un Message Broker suministra la habilidad de encaminar y manipular los mensajes de los sistemas, de forma inteligente. Por ejemplo, un Message Broker podría recibir pedidos de compra de una aplicación Web y encaminar esos pedidos a uno o más sistemas, tomando en consideración una información de pedido. Un Message Broker, también, podría incluir un mecanismo de transformación de información. De esta forma, la transformación seria hecha en un Message Broker y no en los adaptadores específicos de cada tipo de sistema.

Un beneficio de utilización de un Message Broker es simplificar la construcción de interfaces adaptadores de conexión, debido a que, elimina la necesidad de definir interfaces lógicas de manipulación de mensajes individuales, entre cada par de sistemas independientes que están siendo integrados. De esta forma, si se hace una modificación en el flujo de procesamiento de un sistema,

solamente una interface tendría que ser actualizada, aquella en el cual el sistema está siendo modificado, ver Figura 2.6.

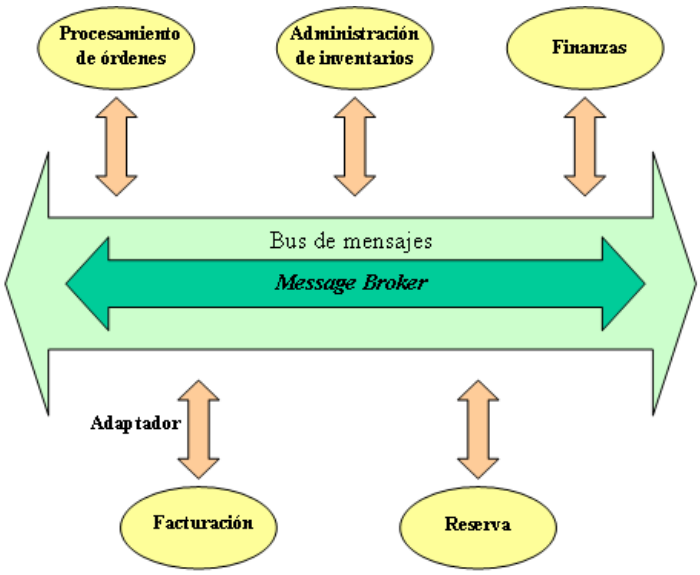


Figura 2.6: Broker de Mensajes

Fuente: [HM 1996].

2.3 Estrategias de integración de sistemas heredados

2.3.1 Redesarrollo

Esta estrategia implica la concepción y desenvolvimiento de un nuevo sistema utilizando nuevas plataformas de hardware y arquitecturas modernas, para sustituir totalmente al Sistema Heredado existente.

2.3.2 Redesarrollo progresivo por módulos

Esta estrategia propone el cambio gradual de los componentes del Sistema Heredado por otros componentes desenvueltos con nuevas tecnologías, preservando los datos y funcionalidades organizacionales del sistema.

2.3.3 Construcción de interfaces modernas mediante Wrapping

Esta técnica de integración consiste en la construcción de nuevas interfaces de acceso, a partir de la adaptación de los componentes existentes que se quieren reutilizar, dichas interfaces se denominan “Wrappers”.

Un Wrapper se puede definir como un modulo de software que acompaña a otros recursos u otros módulos de software con el propósito de otorgarles mejor compatibilidad, adaptabilidad y seguridad.

La idea de mapear el código heredado para ser reutilizado, es un concepto que fue introducido por primera vez por W. Dietrich en 1989 [DWL 1989] y desde entonces se ha ido ampliando y difundiendo. Actualmente, la necesidad de incorporar a Internet servicios proporcionados por sistemas heredados, conjuntamente con la mayor difusión de Corba, J2EE y COM, ha impulsado notoriamente esta idea.

Mediante Wrapping es posible adaptar el sistema heredado para que sea parte de una nueva generación de sistemas, sin los riesgos inherentes a la reescritura y los altos costos de nuevos desarrollos. Es evidente que para que esto no introduzca problemas a futuro, el estado de salud del sistema debe ser "sano" o debe ser factible pasarlo a dicho estado.

Otro de los aspectos importantes de esta técnica, es que permite transformar o adaptar partes de un sistema según las necesidades y las urgencias, manteniendo el resto inalterado.

Mowbray, T. Zahari [MT 1994] describe seis formas diferentes de Wrappers:

- **Layering**; Consiste en construir capas que se complementan en funcionalidad en forma de API y reutilizan el sistema heredado usando un modelo de caja negra.
- **Data Migration**; Convertir o mapear un modelo de representación de datos en otro.
- **Middleware**; Es el caso de los Wrappers para base de datos, que proveen conversión entre diversos formatos y operaciones que se expresan con dialectos diferentes. Un ejemplo es, que programas escritos para trabajar con registros operen sobre base de datos relacionales.
- **Encapsulation**; Separa interfaz de implementación. Puede ser utilizado para encapsular operaciones de programas a los que no se puede modificar su código.
- **Wrappers for Architecture Implementation**.

- **Wrappers for Mediators and Brokers**.

Estas formas de "Wrappers", según la misma referencia, utilizan siete tipos de técnicas diferentes:

- Wrapping vía remote procedure calls.
- Wrapping vía files.
- Wrapping vía sockets.
- Wrapping vía application program interface.
- Wrapping vía events.
- Wrapping vía shared memory.
- Wrapping vía macros.

Algunas veces para la utilización de esta técnica se requieren algunas adaptaciones al código legado o heredado existente, lo cual significa escribir pequeños segmentos de código nuevo. En este proceso se deber poner especial cuidado en escribir el código, de forma tal que, no se generen dos versiones de un mismo programa. Hay varias técnicas de programación que permiten darle un comportamiento dual a un programa para que se comporte según la interfaz desde la que es invocado.

Es también obvio que hay que realizar una serie de pruebas y verificaciones de corrección que involucran: testing individual de los diferentes componentes que intervienen y que han sido adaptados (código legado) y escritos (Wrappers), test de conectividad y finalmente test de integración [HM 1998].

2.3.3.1 Niveles de Adaptación de un Wrapper

Según la clasificación presentada por Harry Sneed [HM 1996], un Wrapper presenta diferentes niveles de adaptación para sistemas heredados, estos niveles se presentan de la siguiente manera:

- **Nivel de Adaptación de Procesos/Jobs**. Un Job corresponde a un proceso Batch, que toma uno o más archivos de Input y produce un resultado sobre un archivo de Output, ver Figura 2.7.

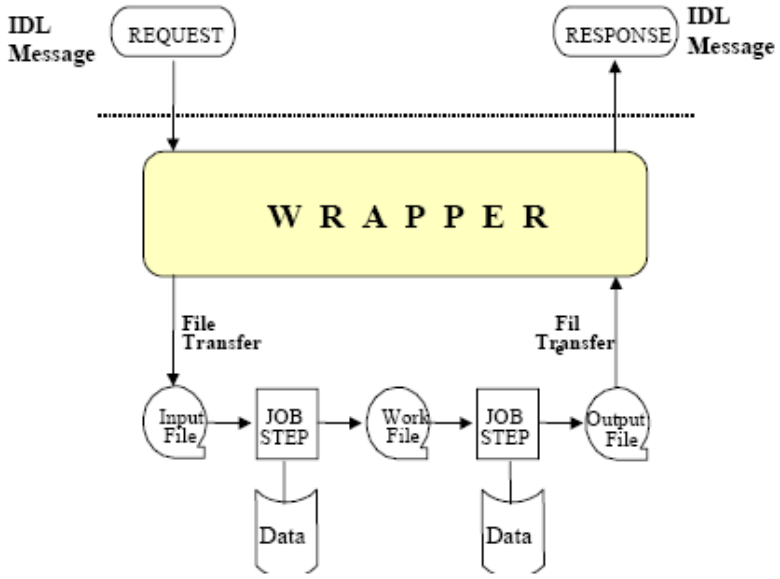


Figura 2.7: Adaptación de Jobs.

Fuente: [HM 1996].

- **Nivel de Adaptación de Transacción.** Una transacción es una operación online invocada desde una Terminal, al enviar un Input Map y finaliza cuando la Terminal recibe un Output Map. Un Wrapper deberá simular la Terminal del usuario, ver Figura 2.8.

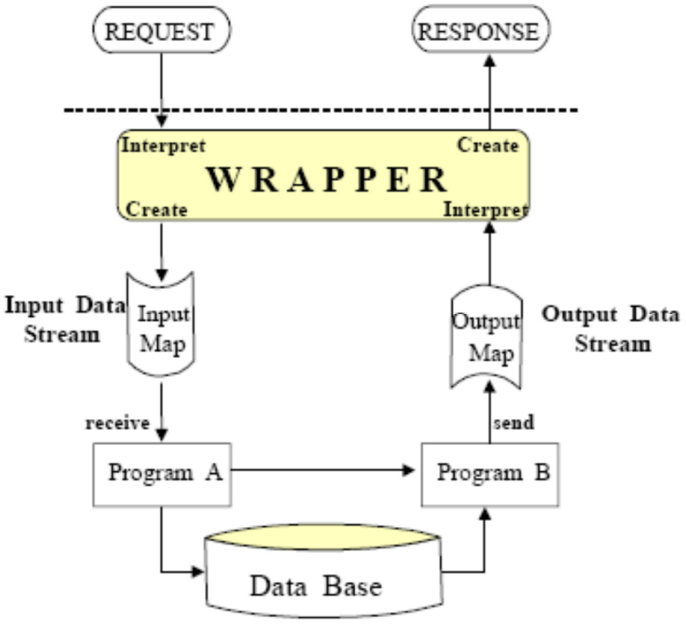


Figura 2.8: Adaptación de Transacciones.

Fuente: [HM 1996].

- **Nivel de Adaptación de Programas.** Un programa es una unidad de proceso que se dispara mediante un procedimiento de control el cual le pasa los nombres de los archivos de input y output. Es similar al Job. No necesariamente es un proceso batch y puede interesar adaptar parte de sus funcionalidades. Este programa no tiene interfaz de usuario, ni tiene interfaz de sistemas, en consecuencia habrá que hacer algunos cambios en el código para poder adaptarlo, ver figura 2.9.

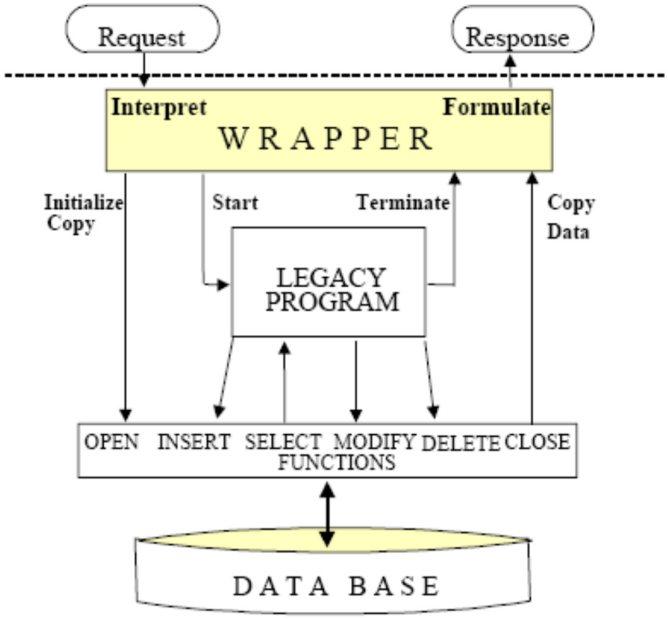


Figura 2.9: Adaptación de Programas.

Fuente: [HM 1996].

- **Nivel de Adaptación de Módulos.** Un módulo es una pieza de software diseñada para ser invocada por otras. En consecuencia tiene su interfaz de sistemas definida. El Wrapper, adapta la interfaz a los requerimientos de arquitectura del sistema e invoca al módulo, ver Figura 2.10.

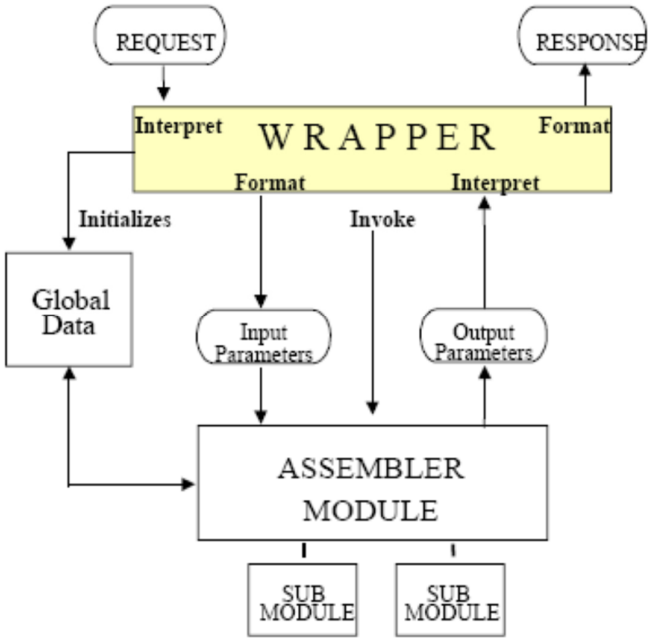


Figura 2.10: Adaptación de Módulos.

Fuente: [HM 1996].

- **Nivel de Adaptación de Procedimientos.** Un procedimiento es una función codificada dentro de un programa, a la cual se la pretende invocar en forma individual. Este tipo de funciones no tiene interfaz propia y por lo tanto hay que modificar el código fuente. Este tipo de modificación no es menor, no porque vaya a modificar la lógica del programa o de la función, sino porque implica un mayor grado de entendimiento del programa y está fuertemente afectado por la documentación existente, ver Figura 2.11.

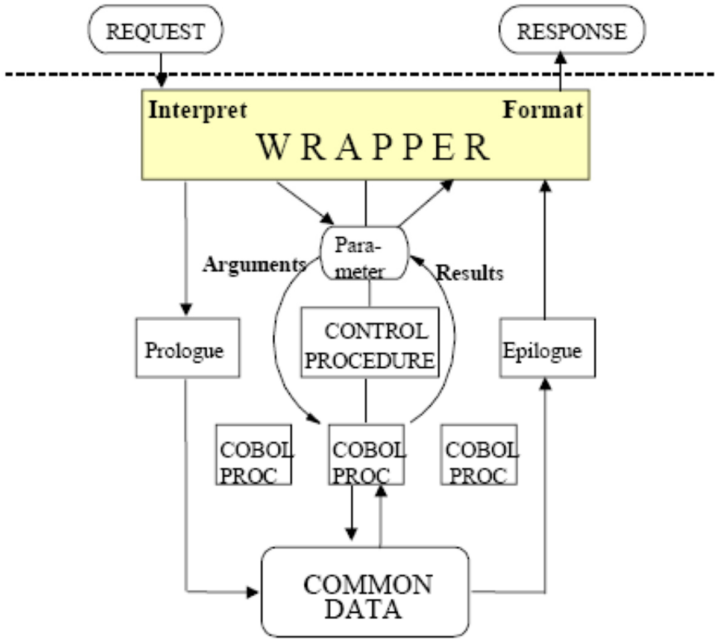


Figura 2.11: Adaptación de Procedimientos.

Fuente: [HM 1996].

2.3.3.2 Encapsulamiento simple y múltiple del Wrapping

Según Paul Asman [PA 2008] existen dos maneras de aplicar los Wrappers en un sistema heredado; este se basa en que podemos encapsular una sistema heredado reemplazando los servicios que queremos exponer incrementalmente; quiere decir que podemos Wrappear toda la aplicación, o Wrappear servicio por servicio.

Desde este contexto un Wrapper forma parte de un sistema orientado a objetos que realizada llamadas a API's de sistemas heredados.

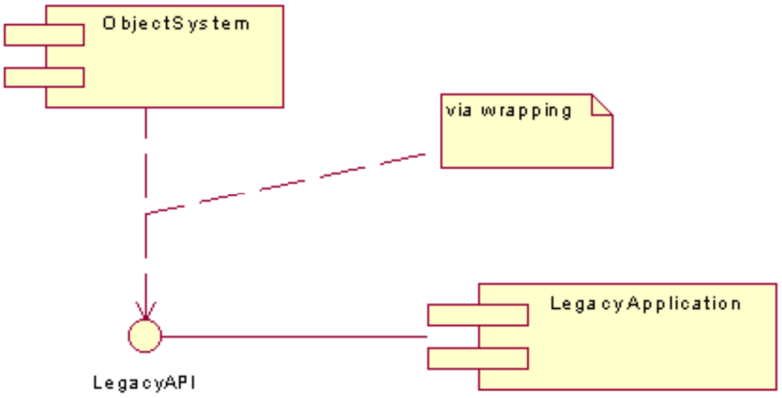


Figura 2.12: Proceso de Wrapping.

Fuente: [PA 2008].

En la Figura 2.12 no se define exactamente si se debe realizar un Wrapping completo de un sistema heredado, encapsulando de manera total todos sus servicios o si deben existir múltiples Wrappers por cada servicio existente en el sistema heredado de manera individual.

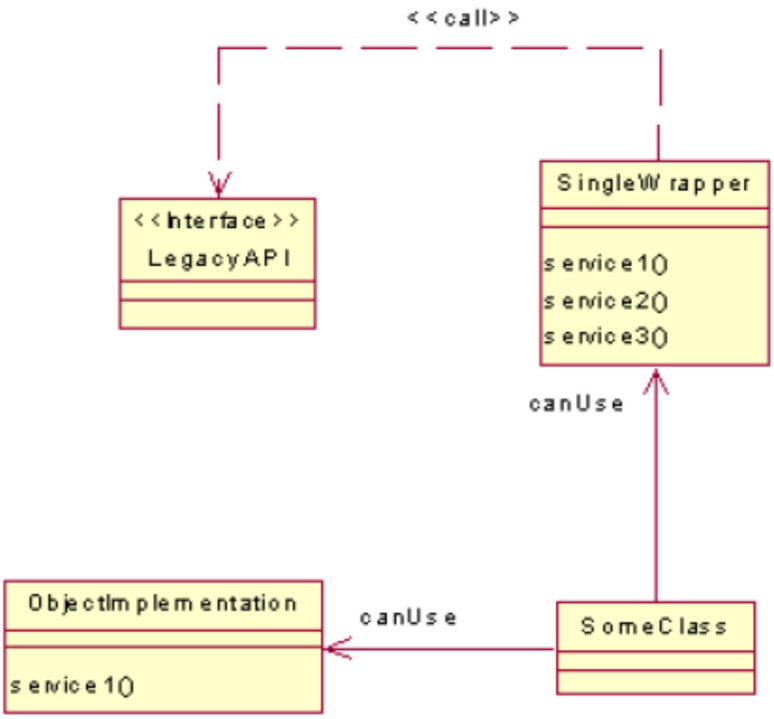


Figura 2.13: Proceso de Wrapping Simple.

Fuente: [PA 2008].

Un Wrapper puede encapsular la lógica de un sistema heredado para su uso

en un sistema más moderno orientado a objetos como se muestra en la Figura 2.13, en este modelo un Wrapping contiene todos los métodos que se invocan desde el API heredado. Cuando un objeto requiere un recurso del sistema heredado envía un mensaje al Wrapper y este realiza la solicitud al API.

Cuando una implementación de algún servicio está disponible (servicio 1), el sistema puede enviar un mensaje a la implementación de dicho servicio dentro del Wrapper.

Cabe resaltar que un Wrapper total o parcial no implementa el API de un sistema legado. El Wrapper realiza llamadas al API, esto está señalado por el estereotipo de dependencia que se resalta en la figura 2.14. El Sistema heredado es quien implementa el API más no un objeto del sistema. En la Figura 2.14 se muestra un modelo incorrecto de un Wrapper.

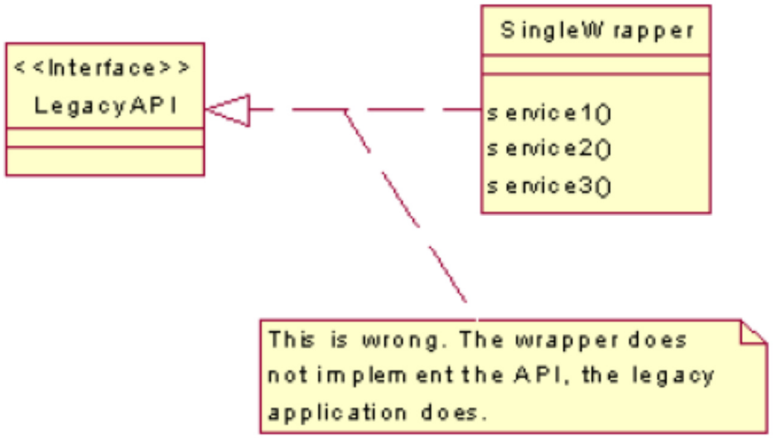


Figura 2.14: Modelo incorrecto de Wrapping.

Fuente: [PA 2008].

Una alternativa a un Wrapping simple es la construcción de múltiples Wrapping, la cual consiste en encapsular cada uno de los servicios que brinda un sistema heredado, véase la Figura 2.15. En este modelo cada miembro de un conjunto de componentes Wrappers encapsula algún servicio que brinda el sistema heredado, y cada uno realiza la llamada al API del sistema heredado. Cuando una nueva implementación de un servicio se encuentra disponible, un objeto del sistema puede usar esta nueva implementación. El correspondiente Wrapper pasa a ser obsoleto una vez que deja de ser utilizado. El objeto puede seguir utilizando al

componente que contiene los Wrappers cada vez que una implementación esté disponible.

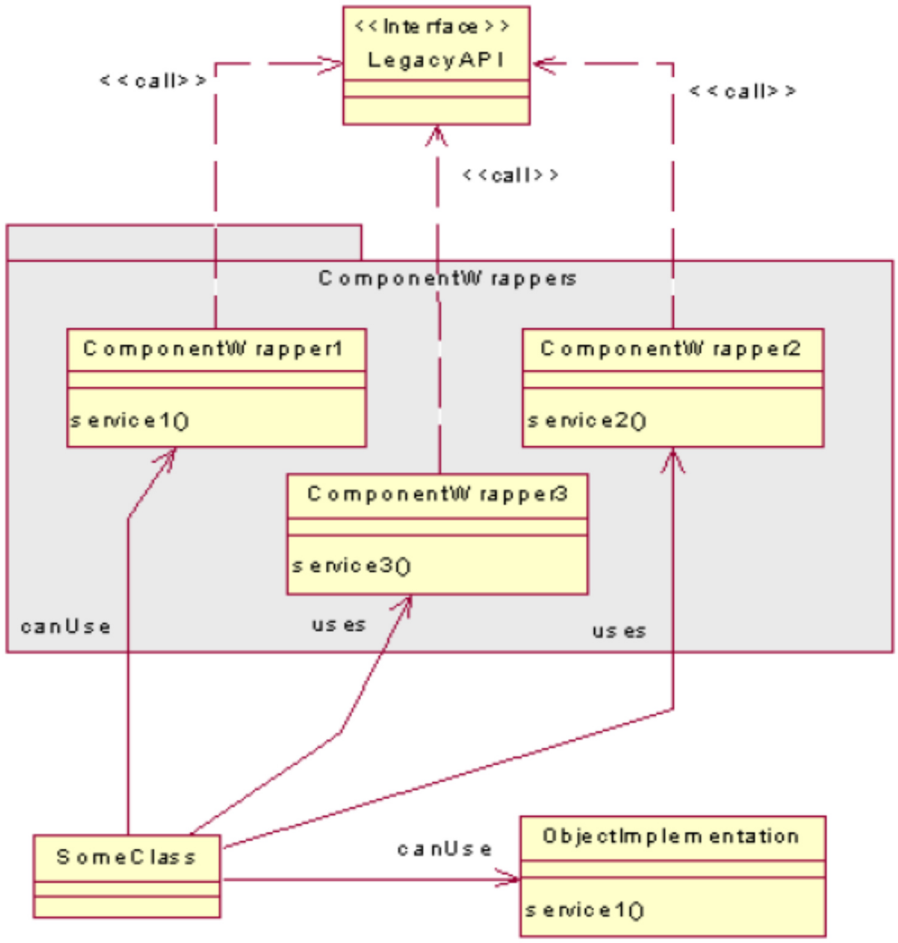


Figura 2.15: Wrapping múltiple.

Fuente: [PA 2008].

Ambos modelos soportan igualmente el reemplazo incremental de sus servicios heredados. En la Figura 2.13 y 2.15 un Wrapper simple o múltiple implementa un servicio mediante un método separado para este servicio.

Cuando un objeto del sistema ya no necesita un servicio este simplemente cambia el método de invocación hacia otro servicio, esto debe cumplirse inclusive si el Wrapper está mal diseñado, ya que el número de Wrappers no hace gran diferencia durante el proceso de integración.

2.3.4 Integración sistemas heredados utilizando Web Services

Los Web Services que utilizan XML y SOAP como elementos principales dentro de su arquitectura, han conseguido encontrar una forma importante de cómo integrar aplicaciones, especialmente cuando estas aplicaciones se encuentran distribuidas.

Según [MI 2008], fundamentalmente existen dos propósitos para implementar un Servicio Web:

Integración de aplicaciones: Se utilizan Servicios Web para el mismo propósito que las actuales y diferentes técnicas de integración, para poder proporcionar integración entre distintas aplicaciones o para proveer acceso a los datos que manejan distintas aplicaciones de diferentes fuentes de datos

Integración de basada en servicios: Consiste en una extensión de la integración de aplicaciones que proveen servicios específicos que pueden incluir agregación, transformación o algún otro valor agregado al servicio que pueda agregar información.

Los Web Services habilitan descentralización y distribución en la integración de servicios.

Las principales consideraciones para usar Servicios Web en la integración de aplicaciones para servicios o procesos de nivel son:

Búsqueda de los servicios: Como hemos definido anteriormente un Servicio Web se publica en un repositorio UDDI. En dicho repositorio se encuentra un documento XML que contiene la especificación de todos los servicios que están disponibles en el servidor. Una manera más precisa es proveer un servidor dedicado UDDI que proporcione un directorio para todos los Web Services que funcionan en los distintos servidores de la organización.

Identificación de la estructura de servicios: El documento WSDL contiene la descripción del servicio que se está ofreciendo, el formato de los datos y los parámetros que necesita el servicio.

Conectar con el servicio: Las peticiones son enviadas al Web Service bajo la forma de un documento SOAP que se transporta sobre HTTP.

El documento SOAP consiste de dos partes la cabecera y el cuerpo, ambos en formato XML, en donde el funcionamiento principal sea una petición del cliente o respuesta del servidor recae sobre el cuerpo del mensaje SOAP.

La cabecera es extensible y puede ser usada para especificar información acerca de la empresa. Debido a que el mensaje SOAP es transportado sobre HTTP, las peticiones pueden pasar sobre un Firewall que filtre y que permita controlar la seguridad del dominio, ver Figura 2.16.

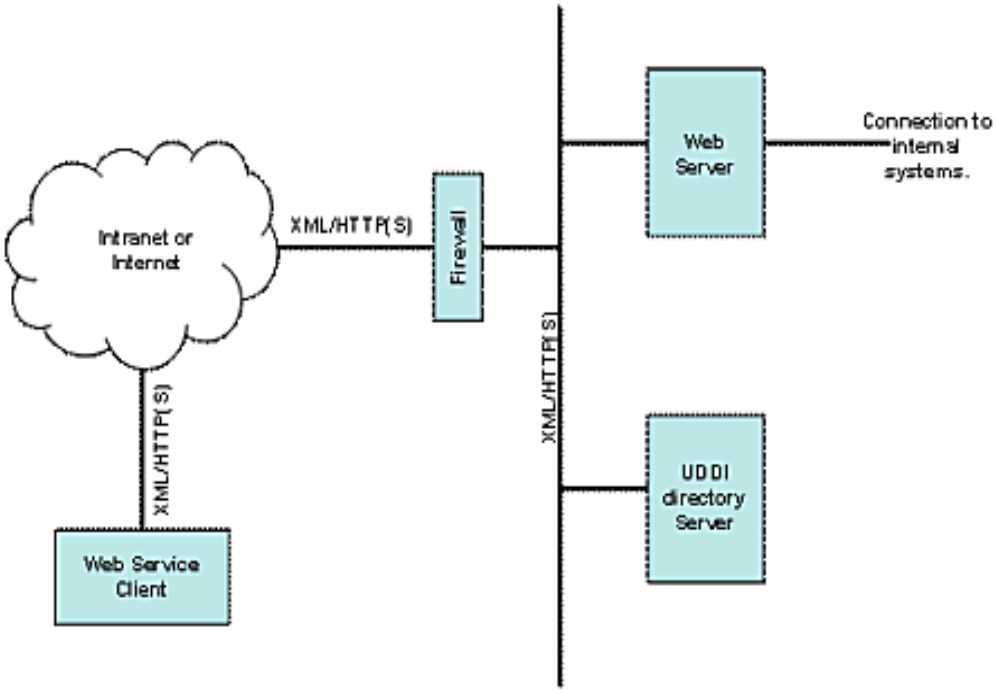


Figura 2.16: Estructura de despliegue de un Servicio Web.

Fuente: [MI 2008].

En general los Servicios Web pueden exponer diferentes tipos de sistemas, dentro de estos encontramos: Bases de datos, sistemas heredados, otros Servicios Web, integración tipo Hub.

En este punto resaltaremos como un Servicio Web puede exponer a un sistema heredado para poder hacer posible su integración.

Según Harry Sneed en [HM 2008], los pasos básicos para poder migrar o exponer un Sistema Heredado a un Servicio Web son los siguientes:

- Recuperar el código heredado.

- Mapear o Wrappear el código que se está tratando de recuperar.
- Hacer que el código esté disponible como Servicio Web.

Para poder recuperar un código heredado, primero es necesario hacer una evaluación de los bloques de código que componen los sistemas heredados y verificar que módulos son necesarios y cuáles de ellos representan operaciones de negocio más importantes, claro está que antes hay que hacer un estudio de factibilidad, a fin de saber si es posible utilizar estos bloques de código.

Sin embargo cabe resaltar que en la mayoría de casos los antiguos sistemas manejan varios bloques de código, de los cuales estos pueden manejar varias operaciones de negocio, resultando una relación "n" a "n" de los bloques de código contra las operaciones de negocio; para solucionar este problema necesitamos hacer un análisis de flujo de datos sobre los resultados finales e identificar todas las declaraciones y así obtener los bloques de código que se encuentran en procedimientos, funciones o subrutinas. Una vez identificado se obtienen las operaciones elementales del negocio y se procede a construir una interfaz común a todas, que permita estandarizar sus tipos de datos y funciones, esta tarea se denomina Wrapping que tal como se explico anteriormente consiste en construir una interfaz moderna de los módulos que se están heredando.

A continuación una vez que los módulos han sido estandarizados mediante el Wrapping se procede a llevar los servicios heredados que proporcionan como Web Services. Entonces a cada servicio heredado y previamente mapeado se le asocia una interfaz WSDL.

La técnica usada es transformar cada entrada en un método y transformar cada parámetro en un elemento de datos de XML. Las estructuras de datos se convertirán en elementos complejos con uno o más sub-elementos. Los métodos tendrán sus discusiones y resultados como referencias a los datos descripciones del elemento. Los métodos y los parámetros serán incorporados a un esquema de XML, ver Figura 2.17.

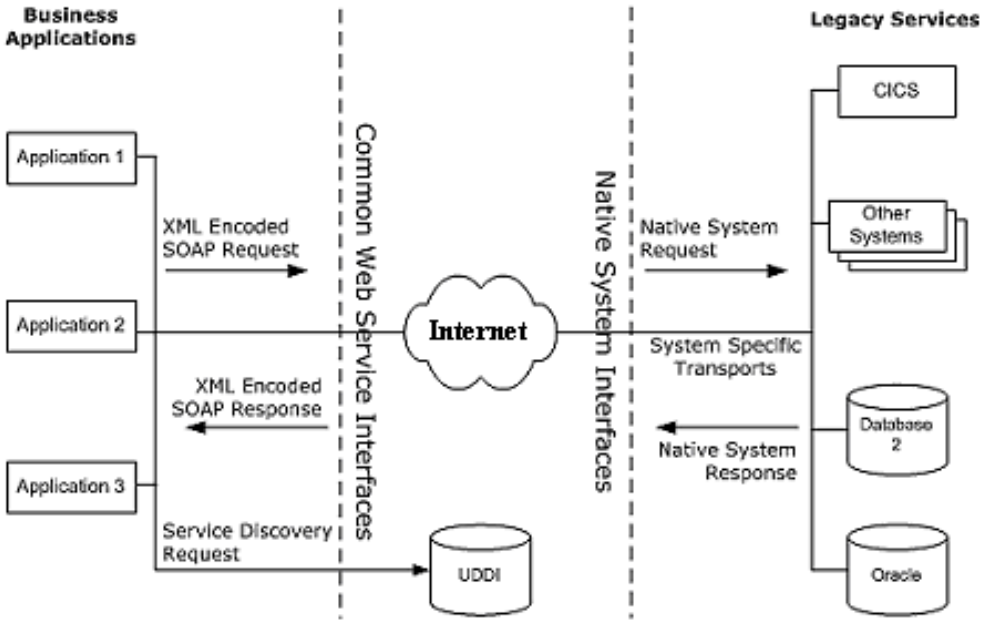


Figura 2.17: Sistemas heredados expuestos como Servicios Web.

Fuente: Creación Propia.

2.4 SOA (Arquitectura Orientada a Servicios)

Con el pasar del tiempos los procesos de negocio han ido evolucionando, en un inicio un proceso de negocio podía ser atendido por la implementación de una sistema, sin embargo esto fue cambiando, actualmente en la mayoría de las organizaciones, para poder atender un proceso del negocio, se requiere la intervención de más de un sistema, presentando de esta forma un esquema de integración horizontal, ver Figura 2.18.

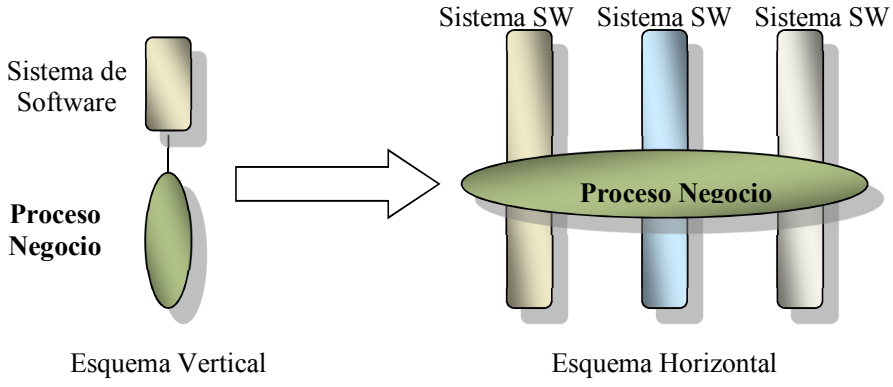


Figura 2.18: Evolución de los Procesos de Negocio.

Fuente: Creación Propia.

Muchas de aplicaciones que se manejan las organizaciones, están desarrolladas en diferentes plataformas, y la integración entre estos Sistemas Heterogéneos representa una tarea muy ardua, la arquitectura SOA aparece como una solución a este problema de integración. El concepto de SOA contempla la funcionalidad de las aplicaciones como servicios, estos servicios vienen a ser bloques fundamentales o entidades lógicas con una funcionalidad definida, estos bloques son variables en tamaño y alcance, ver Figura 2.19.

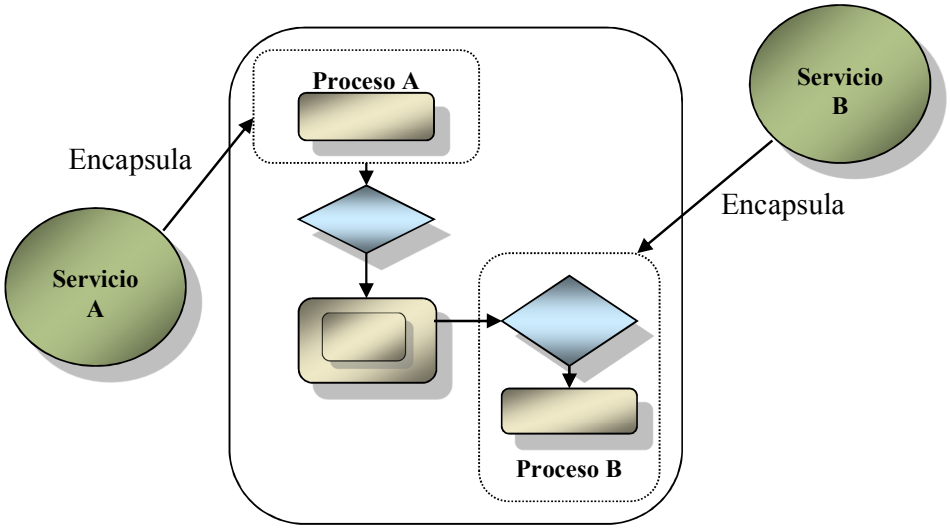


Figura 2.19: Encapsulamiento de los Servicios.

Fuente: Creación Propia.

SOA es una arquitectura que define la utilización de servicios débilmente acoplados y altamente interoperables, para dar soporte a los requerimientos de los usuarios.

La Arquitectura Orientada a Servicios está compuesta por varios elementos funcionales, como son: los protocolos de comunicación entre los consumidores y proveedores de los servicios, la descripción de servicio utilizada para la localización e invocación de los mismos y el registro de servicio utilizado como repositorio, donde el proveedor hace la publicación del servicio y el consumidor realiza la invocación al servicio publicado.

Los servicios en una SOA se relacionan entre sí a través de interfaces que contienen las descripciones de los servicios, en la Figura 2.20, se puede observar cómo se realiza la comunicación en una arquitectura basada en servicios. En los

extremos superior e inferior se encuentran los sistemas que han sido encapsulados en servicios y a través de protocolos de comunicación SOAP se establece la comunicación.

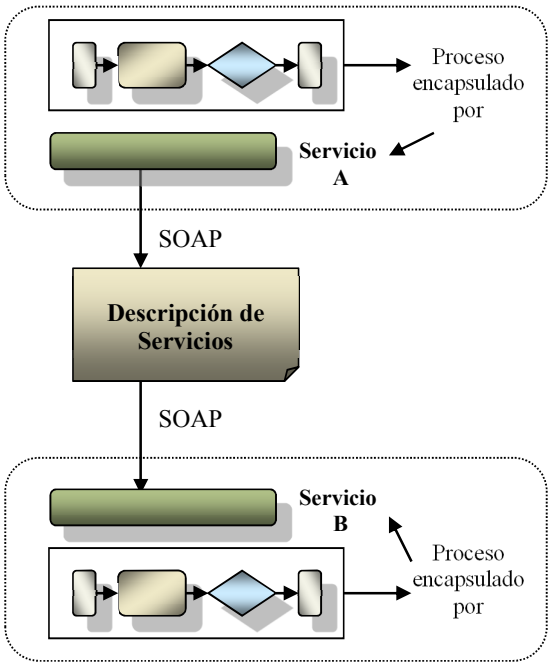


Figura 2.20: comunicación entre servicios a través de mensajes SOAP.

Fuente: [IBM 2004].

Los beneficios de implementar una arquitectura orientada a servicios son los siguientes: [OT 2006].

- Reusabilidad: los servicios que son publicados y expuestos pueden ser reutilizables, reduciendo con ello los tiempos de desarrollo.
- Reducción de costos, la ventaja de reutilizar servicios reduce los tiempos de codificación.
- Escalabilidad: se pueden adicionar más servicios a la arquitectura, e incluso gracias al bajo acoplamiento, los demás servicios no se ven afectados ya que cada uno de ellos opera independientemente.
- Eficiencia al encapsular los procesos de negocios en servicios compartidos.

2.5 Servicios Web

Uno de los inconvenientes del manejo de sistemas lo representa la comunicación entre Sistemas Heterogéneos. Para poder interoperar unos con otros se debería desarrollar un único lenguaje o plataforma y migrar dichas aplicaciones, sin embargo, esta solución representa una labor ardua. Los Servicios Web fueron creados con la finalidad de hacer frente a este tipo de situaciones, ofrecen un medio para exponer y hacer disponibles las funciones de los sistemas, utilizando para ello tecnologías que hacen posible la transmisión de la información.

A continuación presentamos algunas definiciones de Servicios Web:

En [W3C 2004] definen un Servicio Web como "Un componente de software funcional o aplicación Web identificada a través de un URI, cuya interfaz y uso es capaz de ser definida, descrita y descubierta mediante artefactos XML, la cual además soporta interacciones directas con otras aplicaciones de software usando mensajes XML y protocolos basados en Internet".

En [AB 2004] definen al Servicio Web como "Aplicaciones de negocio modulares y auto contenidas que tienen interfaces abiertos, orientados a Internet y basados en interfaces estándares".

En [CS 2005] lo definen como "Una aplicación accesible a otras aplicaciones a través de la Web".

Los Servicios Web encapsulan uno o más aplicaciones ofreciendo una interface única de acceso. Representan una evolución tecnológica, desde el básico modelo cliente-servidor, Middleware distribuidos mediante sistemas basados en RPC (Remote Procedure Call) y Middleware orientados a mensajes (MOM).

Es a partir del concepto de SOA, que nacen los Servicios Web. Estos servicios vienen a ser la implementación de SOA.

Toda arquitectura basada en Servicios Web se rige por roles y elementos operantes, que son mostrados en la Figura 2.21:

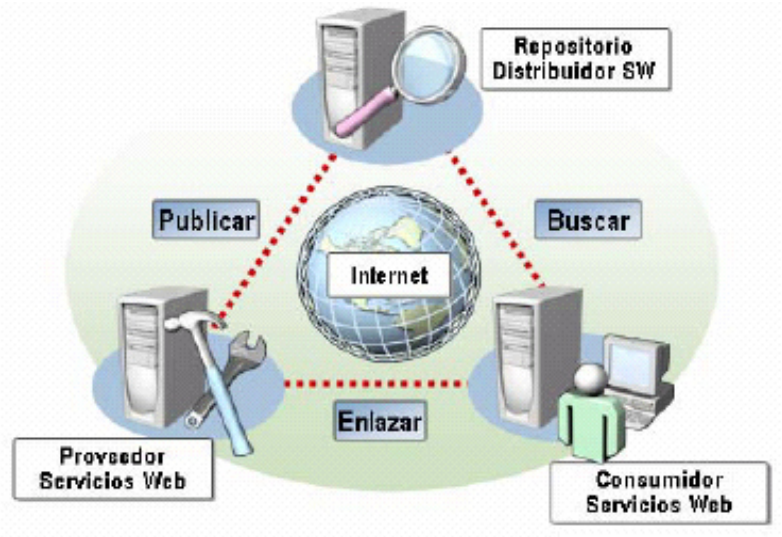


Figura 2.21: Arquitectura de Servicios Web.

Fuente: [OT 2006].

Los elementos que intervienen en una arquitectura basada en Servicios Web son los siguientes:

Proveedor de servicios Web: el proveedor se encarga de la implementación de los servicios y la publicación de los mismos en el repositorio. Este repositorio viene ser el UDDI (Universal Description, Discovery and Integration) que contiene las descripciones de los servicios.

Repositorio de servicios Web: Es el repositorio donde se almacena las descripciones de los servicios publicados por los proveedores. A su vez los repositorios pueden ser accedidos por los consumidores a través de búsquedas, las mismas que devolverán el detalle de las ubicaciones y la descripción de utilización de los servicios.

Cliente o Consumidor de Servicio: es el que solicita el servicio, pero para poder acceder al servicio primero debe hacer una consulta en el repositorio para obtener la descripción del servicio, en la cual se detalla la ubicación y ejecución del servicio deseado.

Las operaciones realizadas con los Servicios Web pueden ocurrir separadamente o iterativamente, y son detallas a continuación:

Publicar/Cancelar: Para ser accesible, una descripción de servicio tiene que ser publicada de modo que el solicitante de servicio pueda encontrarlo, Así mismo

dicha publicación también puede ser cancelada.

Buscar: En la operación de búsqueda, el solicitante del Servicio Web busca la definición del servicio en el repositorio, con la finalidad de obtener los detalles para acceder al servicio deseado.

Enlazar: En la operación de enlazar, se realiza una interacción entre el solicitante de servicio y el proveedor del servicio para lograr el acceso a los servicios.

En [W3C 2004] resumen las principales características de los servicios, en los siguientes puntos:

1. Son accesibles a través de la Web: debido a la utilización de protocolos de transporte como HTTP y la codificación de sus mensajes en SOAP, lenguaje estándar basado en XML.

2. Son interoperables. Debido a que los servicios pueden interactuar independientemente de la plataforma o lenguaje de programación en el cual hayan sido desarrollados.

3. Invocación síncrona y asíncrona: los Servicios Web pueden ofrecer invocaciones tanto síncronas como asíncronas, es decir los solicitantes de los servicios pueden esperar una respuesta a la petición o pueden seguir realizando petición sin la necesidad de auto-bloquearse al esperar la respuesta deseada.

4. Son auto-descriptivos: los Servicios Web guardan las descripciones de sus servicios en interfaces en la cuales se encuentran registrados los datos de localización y mecanismos de ejecución para el acceso a los mismos, estas interfaces son escritas en WSDL (Lenguajes de Descripción de Servicios Web), el cual (al igual que SOAP) es un lenguaje basado en XML.

5. Son localizables: a través de la descripción de los servicios almacenados en los repositorios se puede establecer el enlace y localización de los mismos.

En todo proceso de utilización de Servicios Web intervienen una serie de tecnologías que hacen posible la circulación de información, ver Figura 2.22, los estándares utilizados para el desarrollo de los Servicios Web son XML, WSDL, SOAOP y UDDI, a continuación se entrara a detallar un poco más los conceptos y funcionamiento de cada uno de ellos.

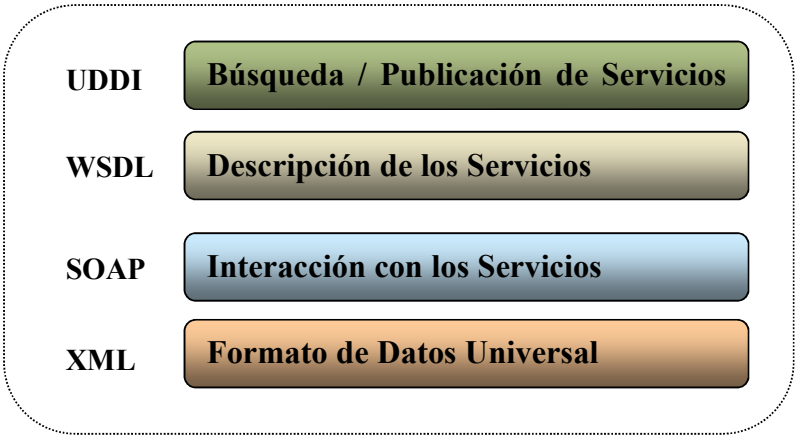


Figura 2.22: Estándares utilizados en una Arquitectura de Servicios Web.

Fuente: Creación Propia

2.5.1 XML

XML (Extensible Markup Language) es un lenguaje de marcas utilizado para estructurar información en un documento o en general en cualquier fichero que contenga texto, como por ejemplo ficheros de configuración de un programa o una tabla de datos [EV 2001].

Es un estándar abierto y libre que fue creado por el consorcio World Wide Web, W3C, los XML almacenan los datos en forma de texto, los mismos que pueden ser leídos desde diferentes aplicaciones soportadas hasta en diferentes plataformas. XML es un metalenguaje, es un lenguaje para definir lenguajes de estructuración de texto basados en marcas explícitas o etiquetas, que son entendibles por una persona y que pueden ser interpretadas por un computador.

Los documentos XML son estructurados, requieren de un elemento raíz y un conjunto de etiquetas, esta estructura interna consta básicamente del uso de cinco caracteres especiales: los símbolos menor que, <, mayor que, >, las comillas dobles, ”, el apostrofe ’ y el carácter &. Los símbolos mayor que y menor son usados como delimitadores de etiquetas, que son los responsables de dar la estructura al documento.

Cada etiqueta está compuesta por un nombre, que a su vez puede contener

uno o más atributos. Los documentos XML pueden ser interpretados de diferentes maneras, es decir se pueden crear etiquetas de más de una forma, pero que al final su interpretación es la misma, esta diversidad en la creación de los documentos XML puede ser uniformizada utilizando documentos de descripción de formatos de datos, estos documentos pueden ser DTD o Schemas XML, la finalidad de ambos es validar los documentos XML para poder uniformizarlos bajo una estructura de datos definida, A continuación se muestra un ejemplo de un documento XML:

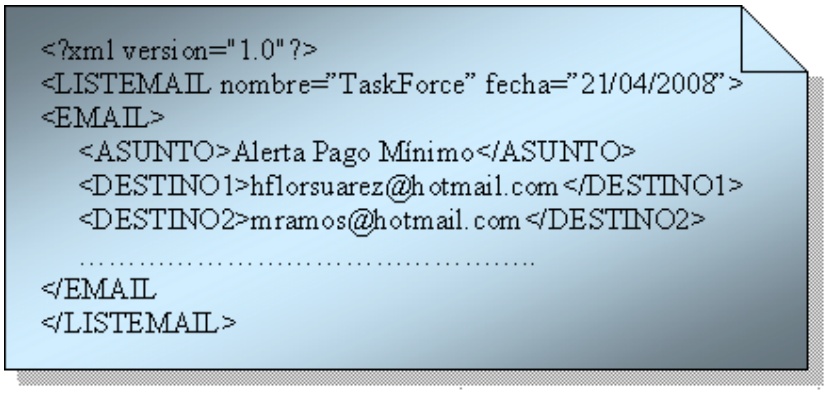


Figura 2.23: Ejemplo de estructura de un documento XML.

Fuente: Creación Propia

En la Figura 2.23 se muestra la estructura de un documento XML, en el se puede observar el manejo de identificadores de documentos y etiquetas personales, este documento es un ejemplo de un XML utilizado para envíos de correos.

2.5.2 SOAP (Simple Object Access Protocol)

SOAP es un protocolo de comunicación entre aplicaciones distribuidas y heterogéneas, la comunicación se basa en envíos de mensajes XML. Fue diseñado para la comunicación vía Internet, lo más característico es que es interoperable e independiente de plataformas y lenguajes en la que se encuentren desarrollados los sistemas a comunicar.

La comunicación en SOAP es unidireccional, es decir existe un emisor que es el cliente y un receptor que viene a ser el servidor. La estructura de los mensajes enviados está compuesta por un conjunto de elementos y atributos, que se encuentran agrupados en tres partes: Envelope, Header y Body. [OT 2006].

- **Envelope:** viene a ser el elemento raíz del documento, es que se encarga de

identificar que el documento XML es un mensaje SOAP.

- **Header:** es el elemento que identifica la información del contenido.
- **Body:** engloba el contenido del mensaje.

SOAP es utilizado también para buscar y localizar Servicios Web que han sido registrados en un repositorio de Descripciones UDDI (Universal Description, Discovery and Integration).

Los protocolos de mensajes pueden ser extensibles, se pueden adicionar funcionalidades a los mensajes, una de las funcionalidades que se pueden adicionar son los Attachments, es decir se puede hacer referencia a otros archivos o imágenes dentro de los mensajes SOAP.

En la Figura 2.24 se muestra un ejemplo de un envió de mensajes SOAP y la estructura de los mensajes, para el caso del ejemplo, el cliente solicita la información de un producto determinado, el precio de venta de una tarjeta de video T-Video 3A, ambos mensajes el enviado y la respuesta mantienen una estructura que está compuesta por un elemento Envelope y un elemento Body.

Figura 2.24: Estructura de mensajes SOAP.

Fuente: Creación Propia

2.5.3 WSDL (Web Service Description Language)

Un WSDL es un documento XML que describe los Servicios Web, en particular sus interfaces [OT 2006]. También es utilizado para describir los mecanismos necesarios para la localización y ejecución de los servicios.

Los proveedores de Servicio utilizan los WSDL para indicar: que métodos pueden ser invocados, que tipos de datos utilizan estos métodos y como acceder a los servicios. Los documentos WSDL vienen a ser el nexo entre los proveedores y consumidores de servicios, ya que el proveedor se encarga de publicar el servicio y el cliente consume el servicio deseado.

Los documentos WSDL describen los Servicios Web utilizando una serie de elementos, ver Figura 2.25:

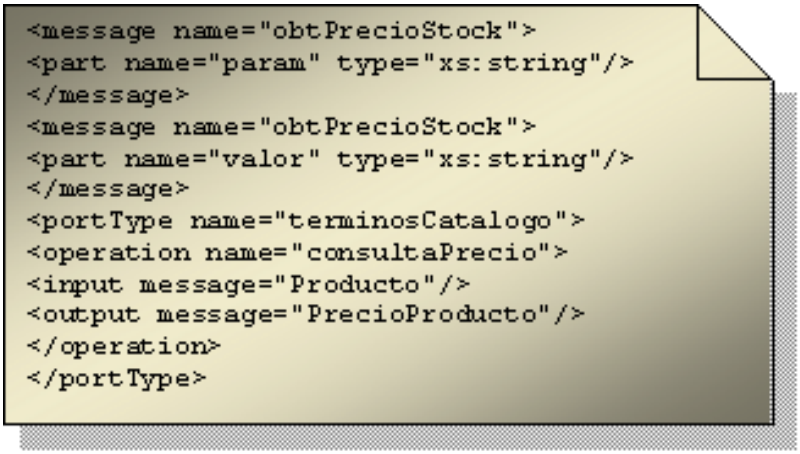


Figura 2.25: Ejemplo de estructura de un documento WSDL.

Fuente: Creación Propia

- **Tipos <types>**: contiene las definiciones de los tipos de datos utilizados por los Servicios Web.
- **Mensaje <Message>**: este elemento contiene el mensaje usado por los servicios, estos contienen los datos que participaran en cada operación.
- **Tipos de Puerto <portType>**: este elemento se encarga de definir todas las operaciones que puede ofrecer el servicio. Los puertos vienen a ser los puntos de conexión a los servicios Web.
- **Enlaces <binding>**: define los protocolos de comunicación utilizados por los Servicios Web.
- **Operaciones**: las operaciones vienen a ser las interacciones

En el ejemplo anterior se muestran los elementos utilizados para descripción de los servicios Web, el nombre "terminosCatalogo" define el puerto utilizado para la invocación del servicio, si hacemos la comparación con un programa clásico, esto sería equivalente a una librería de funciones. También se definió en el documento una operación "consultaPrecio", la cual incluye unos mensajes de entrada y salida "Producto" y "PrecioProducto" que vendrían a ser como los parámetros de input y output respectivamente.

Existen varios tipos de operaciones asociados a los documentos WSDL:

- **Unidireccional**: en este tipo la operación, recibe mensajes, pero no retorna respuesta alguna.

- **Petición - Respuesta**: estas operaciones reciben peticiones y a su vez devuelven una respuesta.
- **Solicitud - Respuesta:** estas operaciones envían peticiones y se quedan en espera a la devolución de una respuesta.

2.5.4 UDDI (Universal Description, Discovery and Integration)

El UDDI es el repositorio en el cual se publica y se buscan los Servicios Web, se encarga de localizar los servicios deseados, esto lo realiza gracias a la descripción de los servicios.

La información contenida en los registros UDDI está compuesta por un conjunto de elementos, almacenados en documentos XML, [AB 2004]:

- **BusinnesEntity**: es el primer elemento en la estructura del documento, describe el negocio, que registro el servicio.
- **BusinnesService**: describir un Servicio Web que ha sido expuesto por una entidad de negocio.
- **BindingTemplate**: este elemento describe la información técnica par el enlazamiento de los Servicios Web.
- **TModel**: La estructura tModel está representada a través de metadatos (datos acerca de los datos). El propósito de un tModel dentro un registro UDDI es proporcionar un sistema de referencia.

La Figura 2.26 muestra el flujo del procesamiento en una arquitectura basada en Servicios Web.

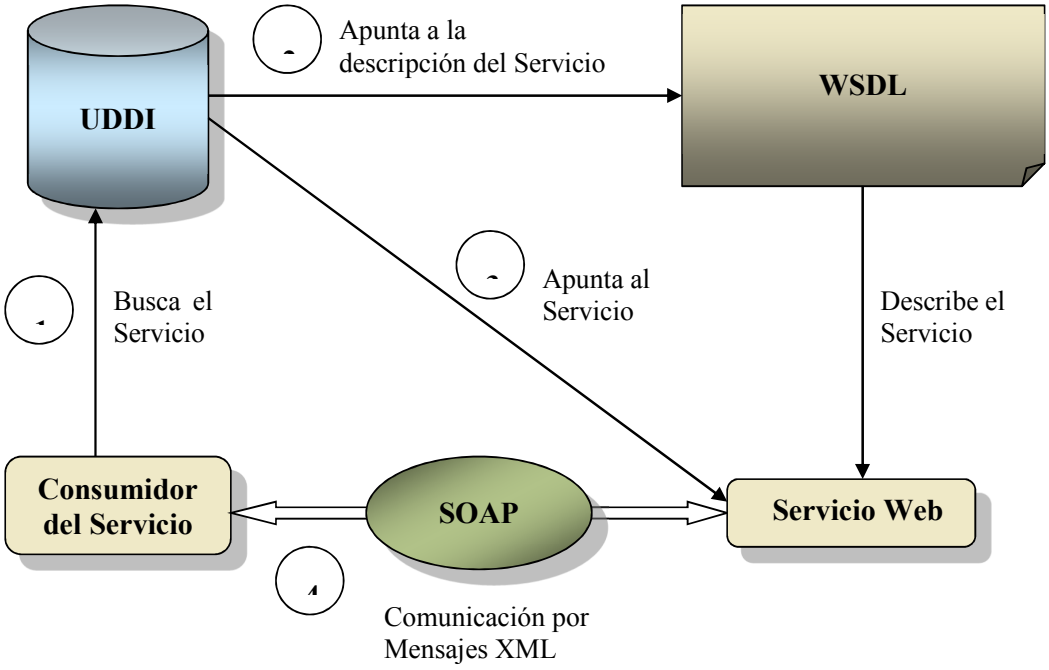


Figura 2.26: Flujo de Proceso en una Arquitectura de Servicios Web.

Fuente: [AE 2007].

La información que proporciona un negocio, a través del manejo de Servicios Web depositados en un repositorio UDDI, consta de 3 partes:

- **Páginas Blancas**: que guarda la información sobre el proveedor del servicio: el nombre, la dirección, el contacto, código postal y otros identificadores, esto es muy similar a la información brindada por los directorios telefónicos.
- **Páginas Amarillas**: que guarda la clasificación de las categorías de los negocios, incluye la categorización industrial, ubicación geográfica y otros identificadores más.
- **Páginas Verdes**: que contienen la información técnica sobre los servicios.

Capítulo 3: Estado del Arte

En este capítulo efectuamos un estudio sobre "el Estado del Arte" de un método para la generación de un Wrapper o conector que nos permita mapear los datos que obtenemos de un sistema heredado a un formato XML, y como este conector puede ser configurado mediante la utilización de una herramienta que ayude a la utilización del conector en la aplicación cliente que lo requiera.

La Herramienta que realiza funciones de soporte a la configuración, tendrá como objetivo principal la configuración de la petición en el Cliente enviando una solicitud en el formato que el Wrapper XML lo requiere, y el Wrapper que es el conector o adaptador tiene como función principal la transformación de los datos que provienen de los sistemas heredados al formato con el que trabaja.

Se publicara el Wrapper como Servicio Web para poder brindar una infraestructura de comunicación para entornos de servicios en el Internet y así lograr interoperabilidad de sistemas.

Al intercambiar ambos módulos documentos XML, se logra un nivel de comunicación estándar sin tener que reescribir el código del sistema heredado que se necesite reutilizar en el desarrollo de un sistema moderno.

3.1 Patrones de Diseño de Software

Es necesario antes definir el patrón de diseño utilizado "Adapters" debido a que la solución se basa en la construcción de adaptadores de software. Uno de los componentes que es parte del método propuesto es el "Wrapper", dicho elemento de software se encarga de adaptar una interfaz a un modo convencional de intercambio de datos, que requiere la aplicación con la que se necesita estandarizar la comunicación.

3.1.1 Adaptadores

Básicamente la función de un adaptador de software o "Adapter", es permitir adaptar una interfaz a otra a fin de que el objeto que es adaptado, pueda colaborar con otro que requiere una interfaz diferente [PA 2008]. Los adaptadores son una pieza fundamental para la interoperabilidad y el reuso de la lógica existente en una

organización.

La adaptación consiste en hacer evolucionar solamente aquellas partes de un sistema que son necesarias para cumplir con requerimientos específicos, sin realizar cambios profundos y sin comprometer el comportamiento y la confiabilidad de los sistemas [PA 2008].

Los casos más típicos para aplicar técnicas de adaptación responden a necesidades de cambios [PA 2008]:

- En la presentación a los usuarios: adaptación de interfaz gráfica.
- En la forma de almacenar la información: adaptación de archivos a bases de datos.
- Necesidad de intercambiar información fluida con otros sistemas: Adaptación por requerimientos de interoperabilidad.
- Adaptación de funciones, para permitir su invocación desde otros ambientes y sistemas: Adaptación de funcionalidades mediante Wrapping (reuso).

Evidentemente, si en el marco de una política de evolución en la cual se marcan los objetivos tecnológicos a alcanzar, se establece un diseño de la arquitectura final deseada y si el estado en el que se encuentra el sistema lo permite, es posible mediante adaptaciones sucesivas evolucionar un sistema. De todas formas, es importante observar que en ninguno de los casos de adaptación que se señalaron anteriormente se menciona la realización de adaptaciones al núcleo central de los sistemas ni ha sido necesario profundizar en el entendimiento del sistema, ya que todas estas adaptaciones se realizan mediante técnicas de caja negra y caja gris valiéndose del uso de adaptadores de software [PA 2008]

En párrafos anteriores a este capítulo se analizó la técnica de adaptación, conocida como “Wrapping”, mientras que en capítulos posteriores se trabajarán específicamente los “Wrappers” vinculados a la extracción de componentes a partir de Sistemas Heredados, concretamente adaptadores para programas y adaptadores para Terminal (interfaz de usuario) [PA 2008], ver Figura 3.1.

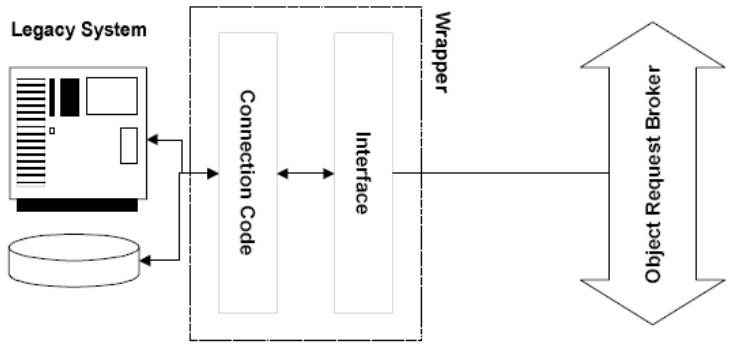


Figura 3.1 Wrapper de integración de Sistemas Heredados

Fuente: [HM 1996].

Para que el adaptador pueda acceder a las funcionalidades de la aplicación se definen interfaces remotas estándares (eje. Web Services), en donde la información de entrada y salida se lleva a cabo en base a formatos de intercambio (eje. XML).

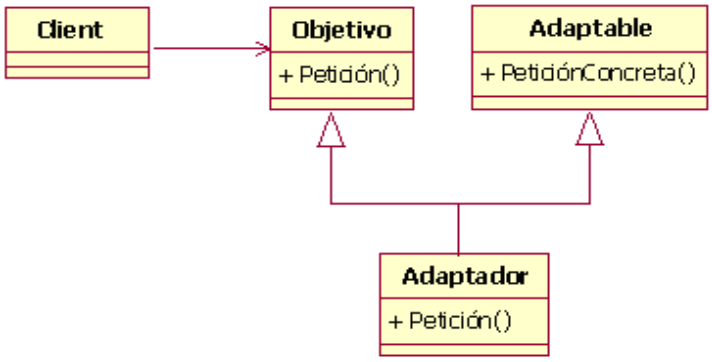


Figura 3.2: Estructura de un Adaptador de Clases.

Fuente: [PA 2008].

- **Objetivo**

  Define la interfaz específica del dominio que usa el Cliente.

- **Cliente**

  Colabora con objetos que se ajustan a la interfaz Objetivo.

- **Adaptable**

  Define una interfaz existente que necesita ser adaptada.

- **Adaptador**

Adapta la interfaz de Adaptable a la interfaz Objetivo.

Existen adaptadores de objetos que a diferencia de el adaptador de clases, el adaptador (Adapter) no hereda de lo que está adaptando, (Ver Figura 3.3).

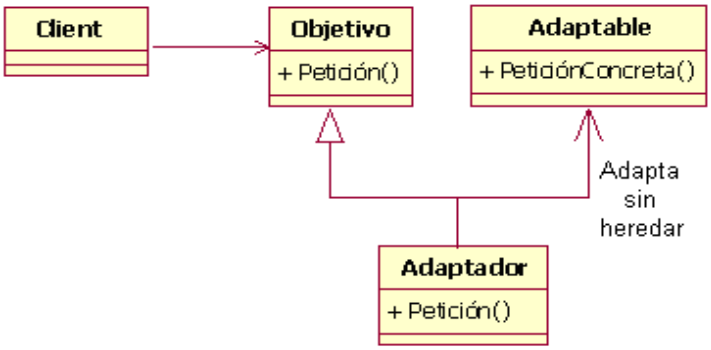


Figura 3.3: Estructura de un Adaptador de Objetos.

Fuente: [PA 2008].

Esta es, sin duda, una técnica que posibilita la construcción de un "Wrapper" de interfaz de usuario a partir de la aplicación de un programa que acepta la entrada/salida de otro programa, la interpreta y extrayendo los elementos importantes al intercambio de información, los pone a disposición de otras aplicaciones, mediante una interfaz de sistema adecuado y público [PA 2008].

Los distintos escenarios en los que se pueden presentar estos sistemas se establecen por combinaciones de elementos tecnológicos que vienen a conformar la arquitectura particular de cada uno. Otros elementos no menos importantes son la diversidad generacional y la calidad de la documentación [PA 2008].

Se presentan dos elementos claros que impactan fuertemente tanto sobre el diseño de la infraestructura que sustente la integración, como sobre los procesos de transformación que puedan ser necesarios para integrar los Sistemas Heredados a un entorno cooperativo [PA 2008].

- **La heterogeneidad,** que requiere del diseño de soluciones particulares a cada caso, así como a la necesidad de recurrir a instrumentos adicionales de conectividad, básicamente Gateways.
- **La documentación**, que puede presentar un obstáculo importante y que debe ser analizada con particular rigurosidad antes de iniciar el proceso de integración y de cuyo estudio deben surgir recomendaciones claras y

procedimientos concretos a fin de llevar la documentación al nivel de certeza necesario para no introducir errores respecto del funcionamiento actual. Existen procedimientos y herramientas de software que permiten combatir algunos de los inconvenientes derivados de la falta de documentación, por ejemplo mediante el uso de adaptadores de software "Wrappers" de caja negra. Asociado a la documentación, pero con relativa independencia de ella, está el entendimiento del programa o sistemas, que tiene que ver con el conocimiento de lo que el sistema o programa efectivamente realiza y con qué finalidad. La asociación de estos elementos, sumado al grado de cumplimiento del sistema y sus programas respecto de las necesidades de los usuarios conforman el estado sanitario de un sistema.

3.2 Clasificación del problema de Construcción y Configuración de Conectores en la Integración de Sistemas

- **Es un Problema de Integración**, pues es el obstáculo principal para las organizaciones que manejan sistemas antiguos que son de alto riesgo operacional dentro del negocio. Este problema se presenta cuando en la organización surge la necesidad de adaptarse a las nuevas tecnologías y nuevos paradigmas en el desarrollo de aplicaciones, que hoy en día están basadas en su mayoría en el uso de la tecnología Web y ofrecen capacidades en entornos distribuidos, que permiten flexibilidad y desacoplamiento en el desarrollo de componentes.
- Otras de las necesidades más urgentes que surgen actualmente dentro de las organizaciones es la de enlazar todos sus procesos de negocios, estos procesos residen en diferentes aplicaciones que cumplen ciertas operaciones, estas aplicaciones por ser criticas para el funcionamiento, son antiguos construidos con tecnología obsoleta y que están ligadas a plataformas de diferentes proveedores.
- Ante ambas necesidades, las organizaciones han tomado la decisión de implementar aplicaciones utilizando tecnología moderna para aprovechar sus capacidades, pero siempre conservando la funcionalidad de sus sistemas antiguos.

- Sin embargo poder utilizar esta tecnología obsoleta desde sistemas de tecnología moderna no es simple, ya que se necesitan implementar estrategias de integración que cumplan dos misiones críticas para la comunicación entre diferentes tipos de sistemas, y son: la construcción de interfaces que adapten y estandaricen los mensajes a intercambiar entre ambos sistemas; y la construcción de interfaces que estandaricen el protocolo de comunicación e intercambio de información.
- **Es un Problema de Implementación**, entre un cliente y un servidor de plataformas heterogéneas que necesiten comunicarse a través de la utilización de una interfaz o conector. Cuando una aplicación cliente necesita implementar servicios que ofrece un sistema heredado, normalmente se invierten tiempos de desarrollo cada vez que se necesite agregar un nuevo servicio o modificar un servicio existente, o en todo caso agregar nuevos adaptadores hacia otros sistemas heredados, lo que hace que el costo de implementar conectores o adaptadores en el cliente sea muy alto, además de que es necesario invertir tiempos para el desarrollo por cada punto de conexión hacia un componente o sistema heredado.
- Según lo mencionado esto quiere decir que mientras se incrementen más puntos de conexión la implementación en la aplicación que necesite utilizar al sistema heredado será cada vez menos flexible; para poder dar solución al problema de las implementaciones de los conectores es necesario contar con un modulo de software que permita agregar dinámicamente adaptadores y agregar y/o modificar nuevos servicios sin la necesidad de tener que codificar continuamente, ahorrando esfuerzos, recursos y tiempos de implementación.

3.3 Método para la Construcción y Configuración de Wrappers XML utilizando Web Services

El presente método describe en particular el intercambio de data entre dos sistemas de distintas plataformas utilizando el lenguaje XML; a través del mapeo de las respuestas que provienen de las consultas a Bases de datos que realizan los componentes heredados.

3.3.1 Razones para la construcción de Conectores Wrappers XML para la integración de Sistemas Heredados

XML es un protocolo estándar para el intercambio de data entre aplicaciones distribuidas o capas de una misma aplicación, y es muy usado hoy en día en entornos Web para envió y recepción de documentos en los que se intercambia datos de manera muy sencilla debido a su flexibilidad y en gran medida a que la mayoría de las plataformas manejan librerías que soportan su implementación.

Por otro lado tenemos sistemas heredados que acceden a fuentes de información (Bases de datos, archivos planos u otras fuentes de información) que en gran mayoría manejan tipos de datos especiales no convencionales que están ligados al fabricante del producto, pues casi siempre obtenemos parámetros de salida de tipos complejos como resultado de las consultas que finalmente serán recepcionados en el sistema que hace uso dichas fuentes de información.

Una de las razones fundamentales por la cual surge la necesidad de intercambiar datos entre sistemas de distintas plataformas es la reusabilidad sobre distintos sistemas antiguos y que son de riesgo critico para el funcionamiento de los procesos dentro de las organizaciones, es por ello que mayormente encontraremos sistemas antiguos que acceden a fuentes de información, y que debido a su seguridad y confiabilidad son difíciles de reemplazar; ante esto surge la necesidad de su reuso en sistemas nuevos que mayormente están implementados sobre entornos Web distribuidos que aprovechan todo el potencial que brinda la tecnología Web.

Debido a esto surge la idea de transformar la información que recogen estos sistemas heredados al formato XML para que puedan intercambiar información con otros sistemas más modernos y de distinta tecnología, sin embargo hay que señalar que la mayoría de los sistemas antiguos no trabajan en entornos Web y tienen una Arquitectura Cliente/Servidor, y aun sí lográramos transformar la información a un formato estándar sería imposible tratar de comunicar un sistema Cliente/servidor con un Sistema Web sin antes llevar la comunicación a una infraestructura común. Para dar solución a este problema se plantea la utilización de Servicios Web que nos permitan llevar estos sistemas heredados a un entorno de Servicios Web y así lograr la interoperabilidad de sistemas.

Reuniendo estas características: transformación de la información utilizando

protocolos estándares, y homogenizando la infraestructura de comunicación a un entorno de Servicios Web podemos lograr la integración de nuestras aplicaciones conservando la funcionalidad antigua que es segura, confiable y sobre todo ahorra tiempos y costos de desarrollo.

Este método de integración puede ser utilizado por un modulo de Software que ayude a configurar las llamadas al conector y los servicios que ofrecen los sistemas heredados, a fin de hacer flexible el mantenimiento del sistema moderno que utilizara funcionalidades del sistema antiguo y así evitar la codificación, ya que dicho modulo nos proporcionara servicios de transformación, configuración y mapeo de la información que proviene en formato XML a un formato que pueda entender el sistema moderno en caso este no trabaje directamente con XML.

A continuación la figura 3.4 muestra la problemática actual a la cual se le aplicara el método propuesto.

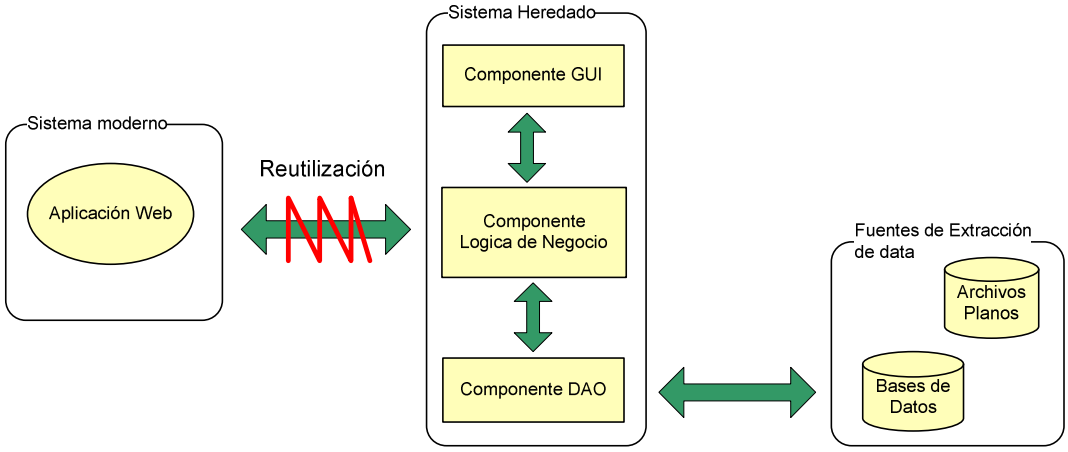


Figura 3.4 Problemática de Integración de Sistemas Heredados

Fuente: Creación Propia.

3.3.2 Descripción del Método para la construcción y configuración de Conectores Wrappers Utilizando Web Services

La presente investigación provee un método y sistema para mapear la información que provienen de los sistemas heredados que manejan tipos de datos complejos y no convencionales a un formato estructurado XML.

El método se basa en identificar los parámetros de salida que devuelven los componentes heredados, y a través de una herramienta que nos ayude a configurar los parámetros de entrada y de salida que solicite el componente heredado, poder realizar llamadas remotas hacia los componentes reutilizables apoyándonos en la transformación de la data a través de un Wrapper y de la exposición del o los componentes heredados en la Web a través de los Servicios Web.

La herramienta o modulo de software realizara tareas de soporte o ayuda (Wizard), en esta herramienta se configurara la llamada a los métodos y/o clases que se quieran heredar del sistema antiguo sin tener que realizar codificación en caso se necesite montar nuevos componentes que se tengan que heredar.

La petición que es generada en la aplicación, que necesita intercambiar datos con algún sistema o componente heredado, envía una solicitud XML que antes es configurada, para ello se registran los parámetros de entrada que están asociados a la solicitud de un servicio "X" que ofrece un componente heredado "X", gracias esto se logra una gran flexibilidad en el manejo de las solicitudes que enviara la aplicación cliente, ya que si por alguna razón se modifica o cambia el componente requerido bastara tan solo con configurar nuevamente la petición.

La petición tiene una estructura definida en formato XML que es reconocida por el Wrapper que mapeará los datos.

Una vez configurada la petición correctamente se procede a enviar invocando remotamente al conector vía la utilización de un Servicio Web que nos permite encontrar al conector Wrapper e instanciarlo. Una vez que la petición es recepcionada el Wrapper realiza una validación del mensaje enviado y obtiene los parámetros de entrada verificando su archivo de configuración XML.

El Wrapper actúa según la petición que se envía ya que esta tiene embebida un ID único con el que se identifica al componente, y es así como puede verificar que parámetros tomara de la petición para invocar al componente requerido; estas validaciones las realiza apoyándose en un archivo de configuración XML en donde se registran todos los componentes a invocar con sus respectivos métodos y parámetros de entrada y salida.

Cuando el Wrapper logra realizar la comunicación con el componente obtiene una respuesta que es verificada antes de ser procesada para saber si se

obtuvo un error al tratar de invocarlo y así devolver una respuesta que pueda entender la aplicación cliente, si la respuesta es correcta el Wrapper procede a mapear la información a un estructura XML definida armando un mensaje de respuesta.

De la misma manera como el Wrapper obtuvo los parámetros de entrada obtiene los parámetros de salida, a fin de verificar si la respuesta contiene estructuras simples o estructuras compuestas (Ej. ResultSets, registros, etc.) y así poder mapear la información de manera correcta.

Una vez que la respuesta es construida, el Wrapper envía el mensaje hacia la aplicación que envió la solicitud, mediante la utilización del Servicio Web invocado que lo instancio de manera remota.

La respuesta es recepcionada específicamente por el modulo de configuración para que procese la información y pueda entregársela a la aplicación.

En la figura 3.5 se muestra el método propuesto para la integración de sistemas heredados usando Wrapping y Servicios Web, apoyados por un modulo que gestione las llamadas al Wrapper o conector y así evitar la constante codificación cada vez que se agregue o modifiquen servicios que ofrezcan los componentes heredados.

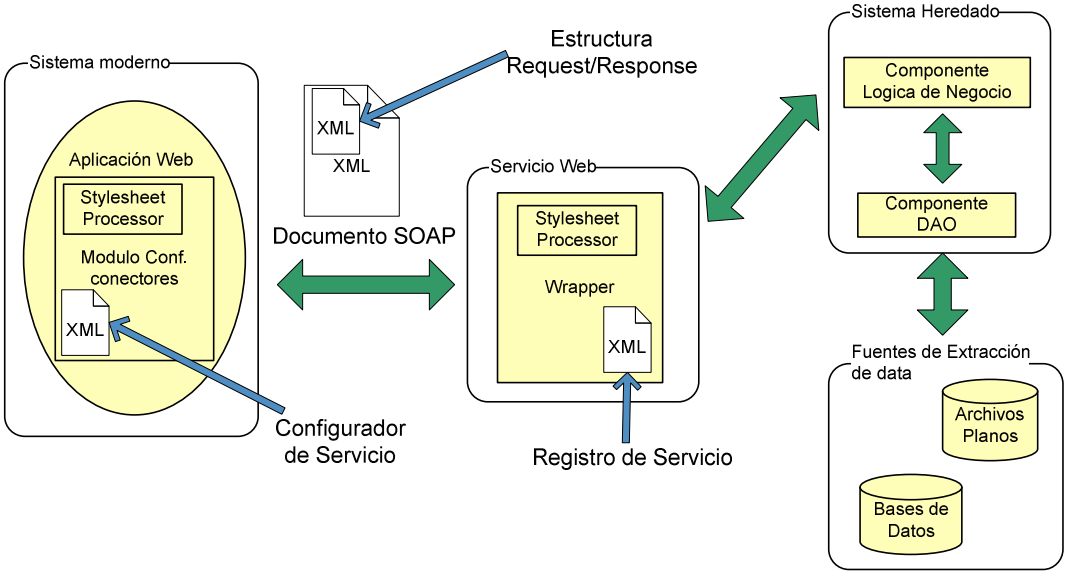


Figura 3.5 Método propuesto para la integración de sistemas heredados

Fuente: Creación Propia.

3.3.3 Restricciones del método para la construcción y configuración de conectores

- Es un modelo en el que los servicios a ofrecer por parte de los sistemas heredados necesitan estar en línea para que pueda ejecutarse las transacciones correctamente, no existe un orquestador de transacciones que controle los mensajes que se envían, ni existen colas que almacenen los mensajes en espera, la comunicación es directa cada vez que se invoca un servicio, si este no está disponible no habrá respuesta.
- El modulo de configuración de conectores solo soporta mensajes de una determinada estructura XML, si los mensajes a intercambiar cumplen con la condición, los conectores o adaptadores y los servicios que se necesiten utilizar podrán ser configurados sin ninguna complejidad de por medio, por el contrario si el mensaje no cumpliera con la estructura definida, antes se tendrá que adaptar a la especificación del mensaje teniendo que codificar para cumplir con dicha especificación.
- Por ser una integración basada en una estrategia punto a punto es imposible manejar un Wrapper (conector a adaptador) genérico para cualquier sistema heredado, en caso de tener que invocar sistemas heredados de distintas tecnologías es necesario utilizar tipos diferentes de Wrappers.
- Cada Wrapper que se necesite a implementar está ligado a una plataforma de desarrollo especifica, por lo tanto cada uno tendrá que construir una lógica diferente para mapear tipos de datos complejos, es decir el algoritmo de transformación a usar por el Wrapper tendrá que variar, aunque mínimamente, parte de su lógica para poder mapear los datos; sin embargo cabe resaltar que el método para la construcción es el mismo y los mensajes a intercambiar no deberían variar en estructura, lo que hace que el modulo de configuración, tampoco sufra cambio alguno ya que el mensaje a intercambiar debería tener la misma forma.
- Una de las restricciones más importantes del modelo es que este solo sirve para poder reutilizar cualquier tipo de sistemas antiguos, para su utilización en la construcción de sistemas basados en tecnologías modernas.

Capítulo 4: Modulo genérico de conectores para la integración de sistemas heredados utilizando wrappers y web services

En capítulos anteriores se detallo las diferentes estrategias de integración de Sistemas Heredados, así como también los conceptos que sirvieron de base para nuestra propuesta de trabajo.

En este capitulo se detallara la arquitectura conceptual de la propuesta, y a su vez se detallara la funcionalidad básica que debe implementar la arquitectura.

4.1 Arquitectura Conceptual del método para la configuración y construcción de Wrappers utilizando Servicios Web

En capítulos anteriores se detallo los conceptos básicos sobre Wrapping y los tipos de encapsulamiento que se pueden realizar, esta técnica es utilizada en nuestra propuesta de trabajo como estrategia de integración de sistemas heredados, la misma que al ser publicada como Servicio Web hace posible el acceso a los servicios que se desean acceder.

Para nuestra propuesta, se pretende desarrollar e implementar una herramienta que permita la integración entre Sistemas Heredados, del lado del cliente (Aplicación Web) y del lado de los servicios a acceder (COM+).

El Wrapper se encargara de mapear la información entre el Cliente Web y el Servicio que se quiere utilizar, sin embargo, para que se pueda establecer la comunicación entre ellos, del lado del cliente, se generara un Cliente de Servicio Web (Proxy) y del lado de los servicios, se creara un Servicio Web que expondrá los métodos del Wrapper.

A continuación se presenta la arquitectura conceptual de nuestra propuesta de trabajo:

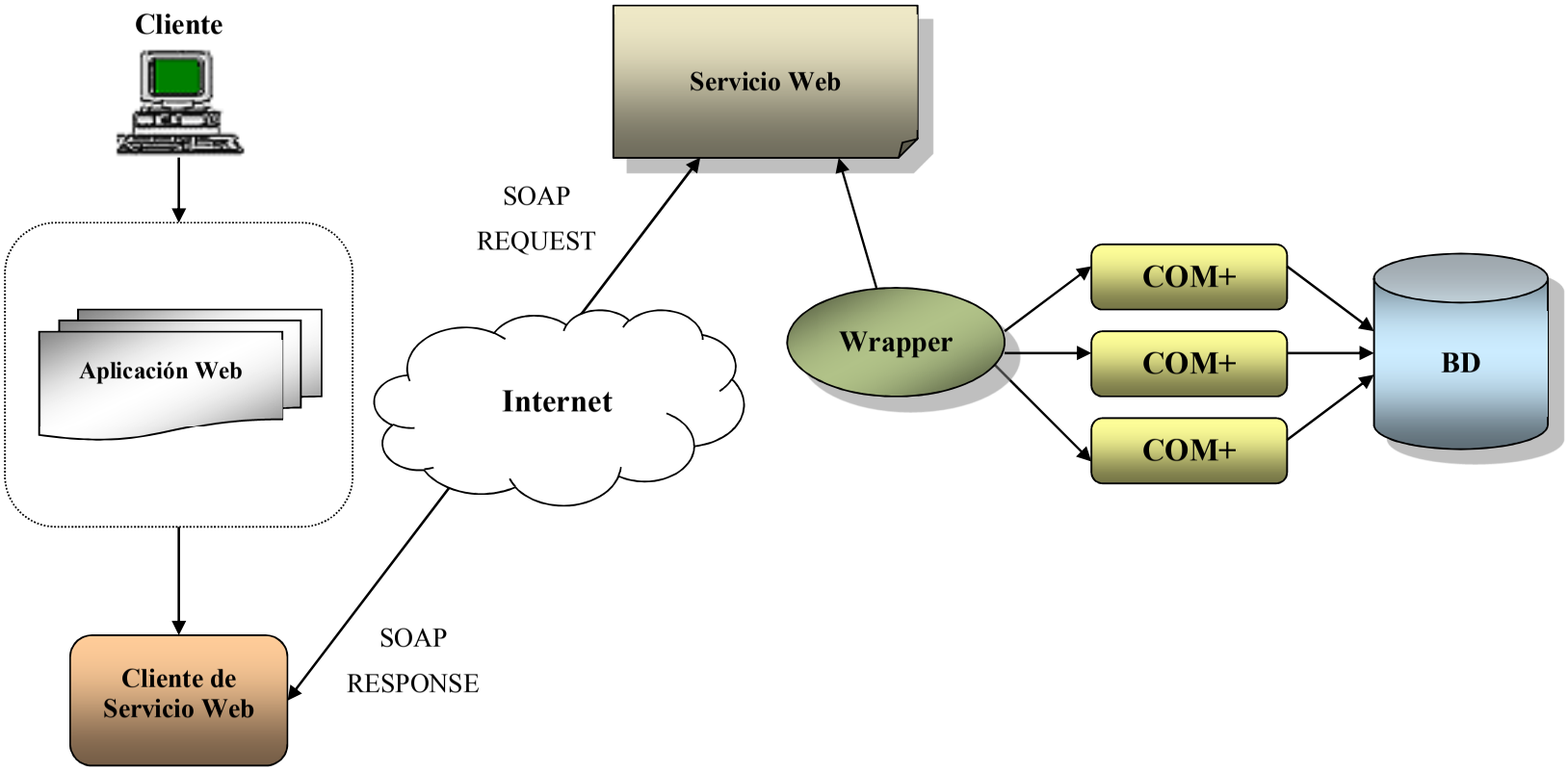


Figura 4.1: Arquitectura Conceptual de Integración de Sistemas Heredados mediante Wrapping.

Fuente: Modelo Propuesto.

En la figura 4.1, el cliente inicia el flujo de comunicación realizando una petición a un servicio, cuya respuesta debe ser obtenida de un componente (COM+), para poder hacer esto posible del lado del cliente se genera un cliente de Servicio Web, que en realidad actúa de conector hacia el Wrapper, este es configurado en el cliente, específicamente en la herramienta que soporta la configuración de distintos conectores , y de lado de los servicios se construye un Wrapper que se encargara de realizar la llamadas a los procedimientos de los componentes deseados. Claro está que el Wrapper ha sido expuesto como servicio web para poder realizar la comunicación hacia el cliente.

A continuación se detallara con más detenimiento cada elemento de la Arquitectura Conceptual.

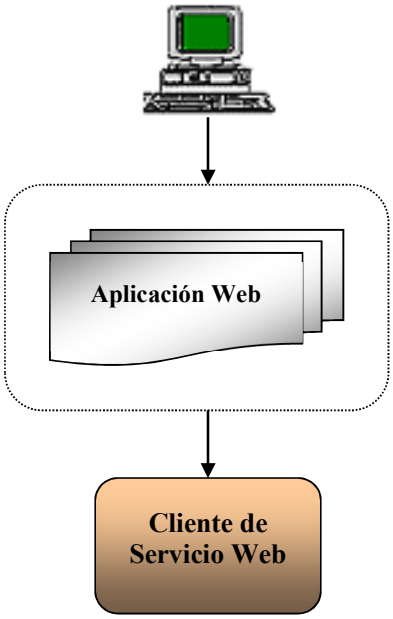


Figura 4.2: Integración de Sistemas mediante Wrapping - Estructura del lado del Cliente.

Fuente: Creación propia

El cliente inicia el flujo realizando una petición para obtener un servicio, para hacer esto posible, del lado del cliente se genera un cliente de Servicio Web, el cual mediante envíos de mensajes SOAP accede a las descripciones de los Servicios ubicados en el repositorio de Servicios.

Cabe señalar que se esta desarrollando, para nuestra propuesta de trabajo, la

interconexión entre Aplicaciones Web y componentes (COM+), esto llevado de una manera muy simple gracias a la configuración y soporte de conectores del lado del cliente, sin embargo queda como futuros trabajos el soporte de distintos conectores para la integración de Aplicaciones Web con otros tipos de Sistemas Heredados, como por ejemplo, sistemas desarrollados en COBOL.

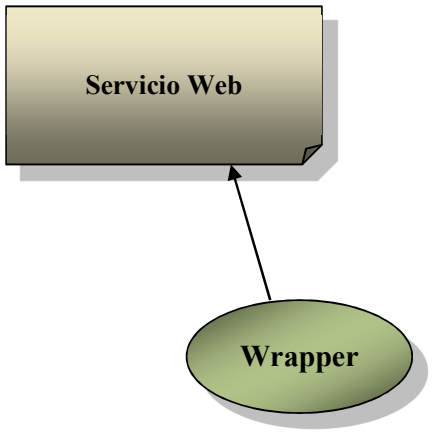


Figura 4.3: Integración de Sistemas mediante Wrapping - Estructura del Wrapper.
Fuente: Creación propia

En la Figura 4.3 se presenta la estructura del Wrapper, el cual es publicado como Servicio Web, con la finalidad de adaptar las funcionalidades de los componentes para luego poder exponerlas en entornos de tipo Internet, de esta forma el cliente de Servicio Web pueda establecer una comunicación mediante el envió de mensajes SOAP para el acceso de los servicios deseados.

El Wrapper se encargara de realizar las llamadas a los servicios adaptados, mapeando la información para poder establecer la comunicación entre los sistemas.

Al ser publicado como Servicio Web, en el Wrapper se configuraran los servicios que se llamaran desde el Cliente Web, así como también los parámetros de entrada y salida por cada transacción a realizar.

En la figura 4.4 se muestra la interacción del Wrapper con los servicios deseados, como se menciono anteriormente el Wrapper se encarga de mapear la información para que pueda ser expuesta, sin embargo estos componentes no necesariamente pueden estar ubicados en el mismo servidor en el que se encuentra el Wrapper, por ello para poder hacer una invocación a los métodos de los componentes se emplearan Llamadas a Procedimientos Remotos, de esta forma el

Wrapper mediante invocaciones podrá acceder a los servicios deseados.

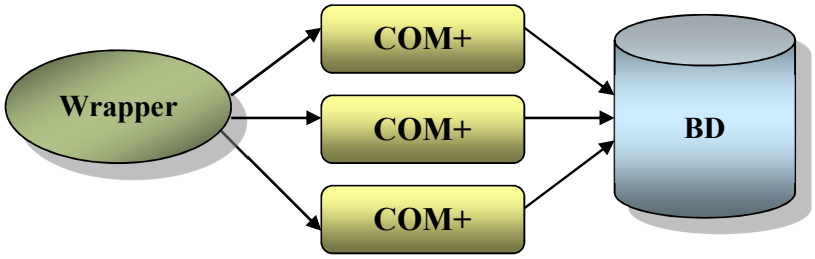


Figura 4.4: Integración de Sistemas mediante Wrapping – Acceso a los componentes.

Fuente: Creación propia

En este tipo de integración de sistemas, utilizando Wrappers, los Sistemas Heredados son adaptados para que puedan ser disponibles entre ellos, no se corren riesgos de reescritura ni de desarrollos adicionales en los mismos, ya que mediante el Wrapping es posible reutilizar las funcionalidades de los Sistemas.

En la Figura 4.5 se muestra la arquitectura de nuestra propuesta de trabajo, en la cual el flujo de comunicación entre el Cliente y los Servicios deseados, parte desde la petición del Cliente, el cual invoca un método remoto del Servicio Web, (Wrapper publicado como Servicio Web), para obtener datos de un servicio determinado (COM+), para ello el cliente envía como parámetros de entrada (Request) una cadena en formato XML.

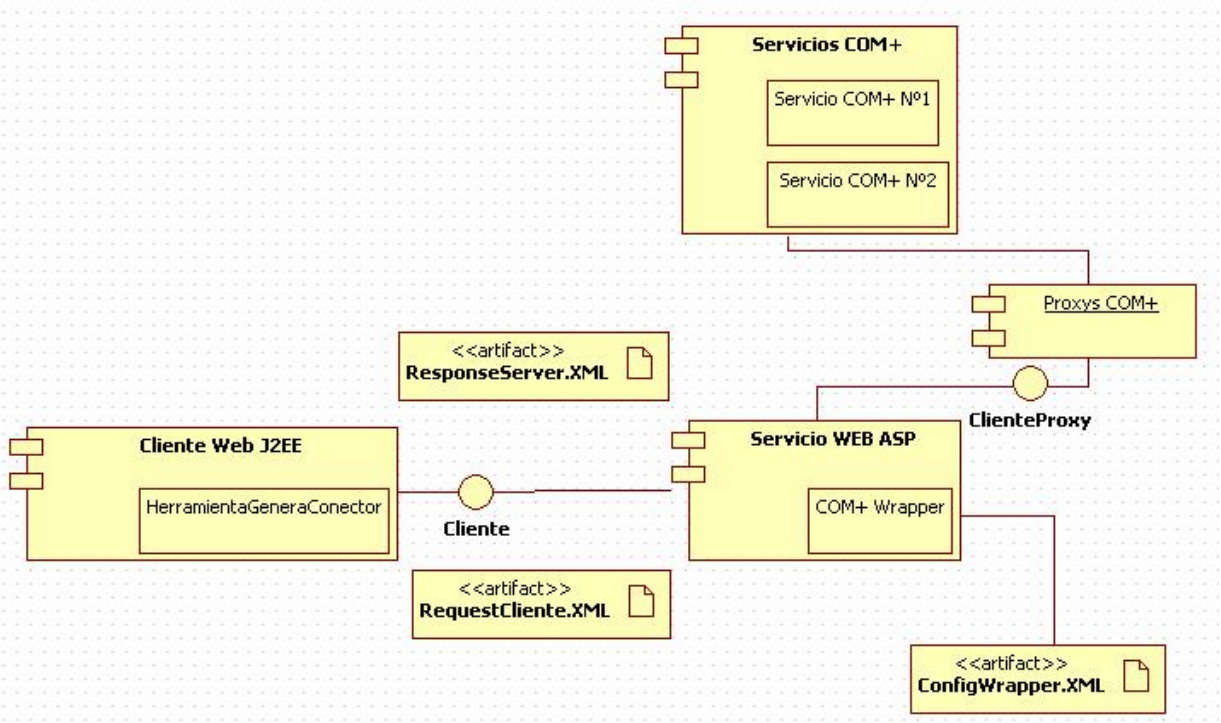


Figura 4.5: Arquitectura de Integración de Sistemas Heredados mediante Wrapping.

Fuente: Creación propia

El Wrapper recibe el dato en formato XML y realiza una verificación en el archivo de configuración del Wrapper (ConfigWrapper.xml), donde esta registrado los servicios a los que se puede acceder, así mismo en el archivo de configuración verifica también, el nombre del COM+ para poder invocar la clase, el método y los parámetros de entrada requeridos.

Una vez realizada la verificación de los datos requeridos para la invocación a los servicios, el Wrapper toma los datos de entrada de la solicitud XML e invoca el COM+ solicitado, vía RPC a través de Proxys.

La respuesta del COM+ es procesada por el Wrapper que verifica los tipos de datos y convierte la respuesta a formato XML, la respuesta es verificada nuevamente en el archivo de configuración del Wrapper y es enviada al cliente.

Por el lado del Cliente, se recibe la respuesta (Response) en formato XML, después de la interpretación y procesamiento del formato recibido, se muestra al Cliente la respuesta a su petición formulada.

Con respecto a la herramienta desarrollada, cabe resaltar que esta nos

permitirá configurar conectores y simplificar grandes desarrollos en la aplicación, en caso se necesiten soportar nuevos conectores, ya que el cliente Web deberá intercambiar siempre mensajes en formato XML sin importar la fuente de donde se tenga que extraer la información, de darse el caso en que el conector no soporte mensajes XML, existirá la posibilidad de construir extensiones para poder formatear el mensaje.

Se construirá un conector, que es el cliente hacia el Wrapper (Servicio Web) y este será montado en la aplicación Web con una simple configuración en los archivos XML que se maneja en la herramienta desarrollada. Para este caso el conector fue construido como parte del desarrollo; sin embargo su implementación es totalmente independiente a la de la aplicación Web, es decir en caso esta necesite cambios en su codificación en la aplicación, no tendremos que realizar modificaciones, también podría afirmarse que si quisiéramos montar conectores hacia otros Sistemas Heredados solo sería necesario configurar un conector, y en caso este no trabaje intercambiando mensajes XML, la herramienta construida deberá soportar la construcción de extensiones que podremos codificar, a fin de que el formato a intercambiar siempre sea XML.

En la figura 4.6 se muestra los módulos de software que son parte de la herramienta construida.

Los módulos Controller, Model y HostTransaction son aquellos se encargan de gestionar la llamada del servicio que se encuentra registrado en el archivo module-config.xml, dentro de sus funciones está la de controlar la llamada de la transacción desde el cliente e identificar el tipo de conector que se está utilizando.

Los módulos AppAnalyzerImpl y el propio conector (Cliente de Servicio Web) son los que se encargan de formatear el mensaje y enviarlo al destino elegido. Si se tuviese un conector que necesite de un adaptador, el modulo que resuelva este problema deberá ser registrado en el archivo AppAction.xml.

El archivo ServiceConfig.xml tienen los parámetros de entrada y salida que solicita el servicio a invocar.

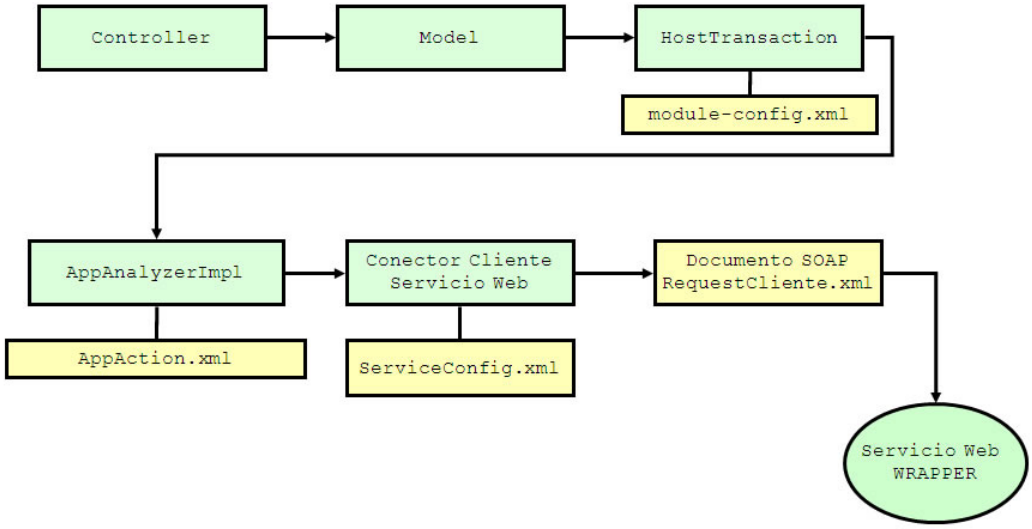


Figura 4.6: Módulos de Software de Herramienta que soporta conectores.

Fuente: Creación propia

En este Capítulo se detallo la arquitectura conceptual de nuestra propuesta de trabajo, en el detallamos todos los elementos involucrados y se explica cómo se realiza la integración de sistemas heredados a través de la utilización de la técnica Wrapping y del uso de Servicios Web; así mismo, también se detalla la gestión de conectores en la aplicación, que hace uso de los Servicios heredados, a través de la construcción de una herramienta que nos ayude a simplificar la complejidad durante el desarrollo.

Capítulo 5: Implementación del módulo genérico de conectores para la integración de sistemas heredados usando wrappers y web services

En anteriores capítulos hemos definido las diferentes estrategias y técnicas de integración de sistemas heredados, se detallo las principales características de cada una de ellas. Posteriormente se procedió a implementar la arquitectura propuesta, resaltando y definiendo las características más importantes de utilizar el Wrapping y el uso de Web Services en función a interfaces basadas en el lenguaje XML que permiten una comunicación más eficiente entre dos sistemas de distintas tecnologías. En este capítulo realizaremos un análisis profundo de los factores más importantes para la implementación de las estrategias, definiendo el problema, planteando su solución, explicando la solución escogida plasmándola en el diseño de una arquitectura física y finalmente se detalla la implementación de dicha arquitectura utilizando las técnicas de integración escogidas.

5.1 Definición del problema

El problema se presenta en un escenario en el cual se necesita desarrollar una aplicación Web la cual debe ser construida sobre tecnología JAVA, dicha aplicación es un sistema de cuentas por pagar para control de proveedores, ingreso de productos y manejo de Kardex en general; sin embargo en la empresa ya existe un sistema antiguo Stand-Alone construido bajo la plataforma VISUAL BASIC 6.0, este sistema está compuesto por un conjunto de componentes donde la interfaces de usuario, lógica de negocio y/o acceso a datos son independientes e interactúan entre sí; el COM que ofrece la funcionalidad de acceso a los datos utiliza una Base de datos SQL Server 2000 y es el componente que se necesita reutilizar para que sea usada por la nueva aplicación Web, ante esta problemática de querer comunicar un COM con la aplicación Web JAVA, surge la necesidad de plantear la utilización de estrategias de integración basadas en la utilización de interfaces estándares que permitan manejar un lenguaje de intercambio común de mensajes entre ambos módulos de software.

5.2 Descripción de la solución

Para dar solución al problema de la comunicación entre una aplicación Web y un componente COM, hay que uniformizar la infraestructura de comunicación entre ambos, para ello necesitamos poder utilizar el COM en la Web utilizando una de las nuevas estrategias de integración como son los Servicios Web, con ello logramos llevar el COM heredado a un entorno de servicios tipo Internet, para que pueda ser publicado en la Web y así poder ser consumido por las nuevas tecnologías que hacen uso de la Internet.

Para dar solución al problema de la uniformidad e incompatibilidad de los tipos de datos que se intercambian por la Web utilizando Servicios Web, se utiliza una estrategia de integración conocida como Wrapping, la cual transforma los tipos de datos complejos que no pueden ser procesados en el Servicio Web a un tipo de dato simple, en este caso por tratarse de un COM, los tipos de datos o funciones no compatibles que podría retornar son: propiedades, parámetros por referencias y objetos RecordSet; el procedimiento para poder transformar estos tipos de datos consiste en formar estructuras XML y enviarlas por la Web, de tal forma que el tipo de dato a intercambiar sea una cadena con formato XML; así mismo la aplicación Web construida en JAVA deberá enviar como petición una cadena, también en formato XML, con los parámetros de entrada que solicite el componente. Es así como el Wrapping actúa como un mapeador de los tipos de datos y funciones del COM, transformándolas a un formato de lenguaje de intercambio común, en este caso XML.

A nivel de la Aplicación Web como parte de la arquitectura planteada, se construyo un modulo de software, que es la herramienta que servirá para poder montar conectores que intercambien mensajes con el sistema heredado. En este modulo de software se configuran y registran las transacciones en archivos XML haciendo flexible la llamada a los servicios que se ofrecen desde el sistema heredado.

El conector debe enviar una petición y recibir una respuesta en formato XML, y es que el modulo de configuración solo soporta conectores que intercambien mensajes en el lenguaje señalado. En caso se necesite montar conectores que no intercambien mensajes XML podremos desarrollar extensiones

y configurarlas en la herramienta (modulo de seguridad), a fin de formatear el mensaje que se envía y recibe en el formato aceptado.

Para el caso planteado la aplicación Wrapper que es llevada a un servicio Web, intercambia mensajes (cadenas en formato XML) con la Aplicación Web, esto se lleva cabo de la siguiente manera: el conector Wrapper (Servicio Web) intercambia mensajes XML enviando documentos SOAP por la Web, estos documentos poseen parámetros de petición y respuesta cuyos tipos de datos son cadenas; cuando la solicitud es enviada desde la Aplicación Web, la herramienta transforma la petición en una cadena con formato XML con una estructura de petición definida y viaja hacia el Wrapper quien entrega la cadena(formato XML) que contiene los parámetros de entrada hacia el COM solicitado, este la procesa y da una respuesta, luego transforma la respuesta a una cadena en formato XML y envía esta cadena a la Aplicación Web, aquí es donde la herramienta construida toma la cadena de respuesta en formato XML y se encarga de transformarla a los tipos de datos que maneja la aplicación Web para que esta pueda presentar los datos en pantalla.

En este caso el conector fue desarrollado como parte de la solución y constituye una pieza fundamental dentro de la arquitectura desarrollada.

El conector es el Wrapper, el cual se encarga de recibir la petición (XML) procesarla, obtener el resultado y enviarla (XML); este método es expuesto, publicando el componente Wrapper como Servicio Web para que pueda conversar con la aplicación Web, utilizando para ello el protocolo HTTP para poder realizar la comunicación.

El Wrapper tiene como tarea invocar el componente y mapear la respuesta, esto se realiza fácilmente debido a que el Wrapper sabe como invocar un componente ya que posee un archivo XML donde se configuran las transacciones, las asociaciones de las transacciones con cada COM, y los parámetros envió/respuesta con el Path XML de donde leerá el valor de cada parámetro según sea la petición XML o la respuesta XML.

De lado del Cliente solo hay que invocar al conector, esto se logro construyendo un Proxy (Cliente de Servicio Web) hacia el conector Servicio Web Wrapper, y se configuro en el modulo genérico de configuración de conectores.

5.3 Arquitectura Física de la Arquitectura planteada

En la arquitectura planteada podemos identificar varios elementos que constituyen los componentes más importantes de la arquitectura, en ella podemos reconocer los siguientes: La **Aplicación Web**, que está construida bajo la plataforma J2EE java se comporta como la aplicación de consumo que invocara el conector Wrapper, es una aplicación moderna que necesita reutilizar los COM heredados para su desarrollo, en ella reside el **modulo o herramienta de configuración genérica de conectores,** en la que se configura el conector que en este caso es el Cliente de Servicio Web " Wrapper".

El Servicio Web Wrapper es el modulo conector que contiene dentro de su lógica al componente COM Wrapper, que al ser publicado como servicio Web permite exponer sus métodos para ser utilizados por cualquier aplicación que haga uso de un entorno tipo Internet. El Wrapper en si, encapsula la lógica que permite mapear o adaptar la funcionalidad de los componentes COM heredados, hacia tipos de datos convencionales que puedan ser expuestos sin ninguna dificultad por el Servicio Web que lo expone.

El COM Wrapper realiza llamadas a los COM heredados para adaptarlos y formatear la respuesta en una estructura XML, estas llamadas puedan ser llamadas a procedimientos remotos RPC de ser el caso que los componentes se encuentren físicamente en otros servidores.

El Servicio Web fue construido a partir del COM Wrapper conector-adaptador, utilizando la herramienta Microsoft SOAP Toolkit que permite publicar COM como Servicios Web, el resultado es un Servicio Web que puede ser generado en ASP o en un paquete ISAPI.

El **COM o los COM** heredados son aplicaciones antiguas que se utilizan en antiguos sistemas de la organización que están construidas bajo la plataforma Visual Basic 6.0 y realizan conexiones hacia SQL Server 2000 para obtener data de las fuentes de información, como se explico este componente contiene la lógica del acceso a datos del sistema de proveedores y control de Kardex, el objetivo principal del desarrollo de esta arquitectura es poder reutilizar estos componentes desde la aplicación Web JAVA y ahorrar recursos, esfuerzos y tiempos en el desarrollo de la nueva aplicación.

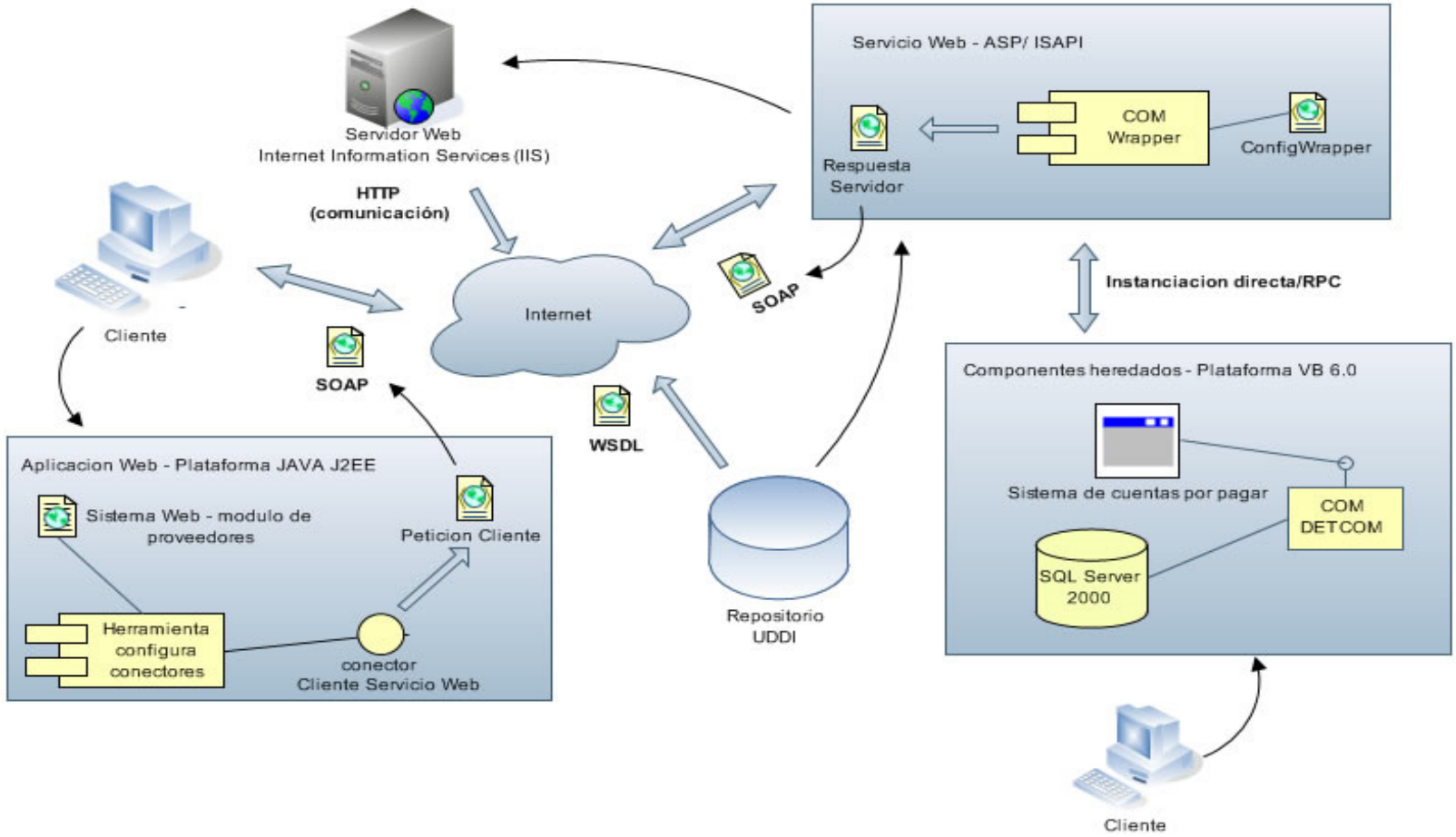


Figura 5.1. Arquitectura Física de Integración de Sistema Heredado Usando Web Services.

Fuente: Modelo Propuesto.

5.4 Implementación de integración de sistemas heredados utilizando Wrappers y Web Services.

En el desarrollo del presente trabajo se trata de demostrar de manera sistematizada que existen estrategias de integración de sistemas heredados que nos permiten integrar sistemas de distintas tecnologías, permitiendo reutilizar sus funcionalidades existentes. Entre las funciones principales esta la adaptación de los servicios que brindan los componentes heredados mediante el uso de la estrategia Wrapping y la solución a los problemas de interoperabilidad entre dos sistemas, que no solo son de tecnologías diferentes sino de distintos entornos de comunicación, mediante el uso de Web Services que se basan en la utilización de protocolos estándares de comunicación de gran aceptación en Internet.

A continuación se muestra el diagrama de flujo con la interacción de todos los módulos de software que participan en la implementación de la arquitectura.

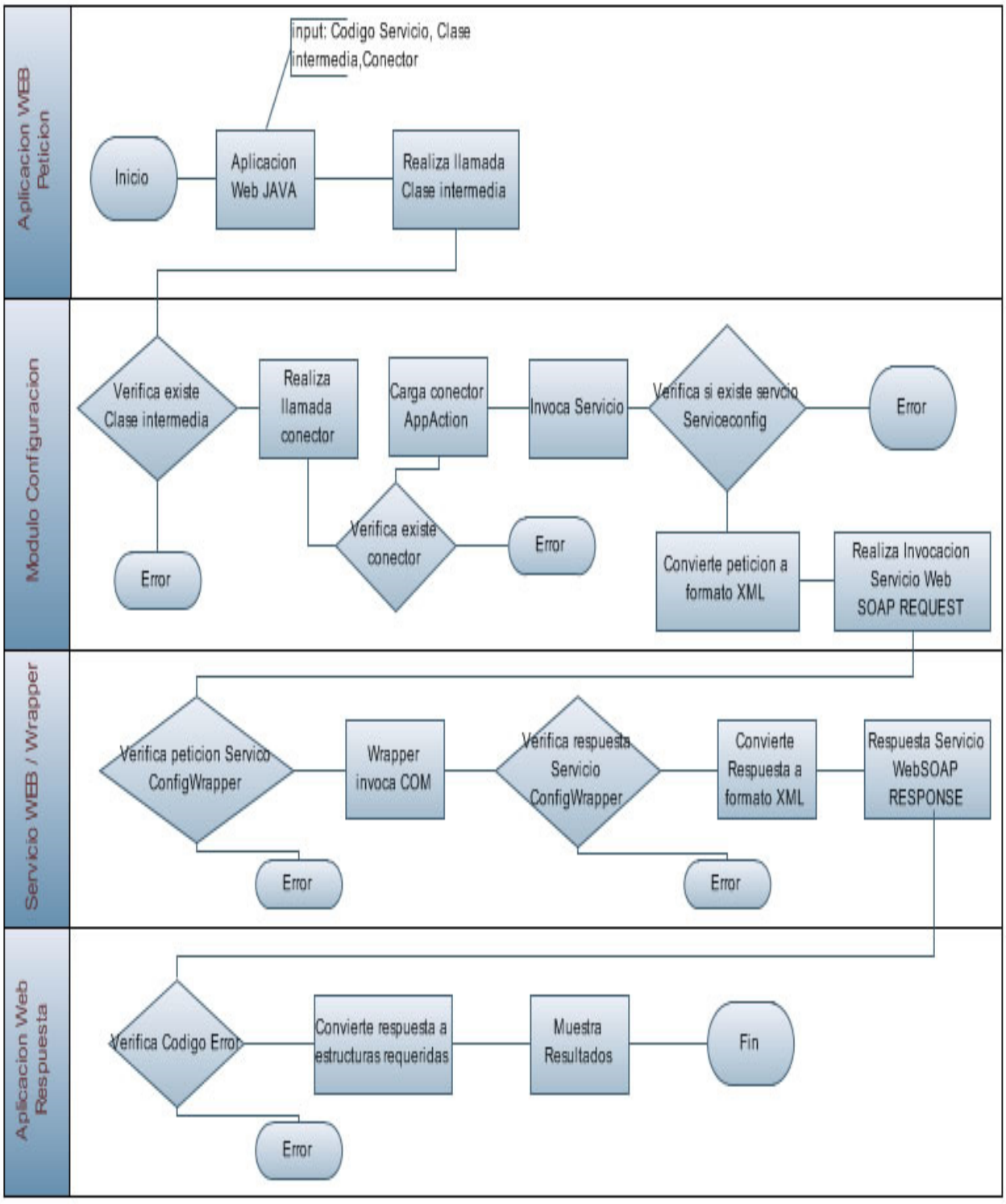


Figura 5.2. Diagrama de Flujo de la integración de sistemas heredados utilizando Wrapping y Web Services.

Fuente: Creación Propia.

Los paquetes o módulos de software de los que se compone la aplicación Detcom son los siguientes:

Aplicación SiteDemo Detcom (Web Application)	Aplicativo que contiene los archivos HTML y JSP (páginas Web y paginas dinámicas) que componen la capa de presentación de toda la aplicación
Módulos TConnector, MConnector y EConnector (Java Application)	Son los módulos que componen la herramienta de configuración genérica de conectores, contienen las Clases que se encargan de la invocación del conector, la configuración del Servicio y de la transformación de la petición a formato XML.
Detcom-Bussines (Java Application)	Contiene las clases intermedias que se utilizan dentro del proyecto y que sirven para poder implementar e instanciar al conector genérico, y así poder agregar funcionalidad a la lógica del modulo conector. Contiene además la lógica de control que reside en los Servlets que se invocan desde cada página JSP, y que realizan llamadas al modulo genérico de conectores.
Web Service WrapperSoap	Es el Servicio Web que expone los métodos del COM Wrapper, para que puedan ser utilizados de manera remota y a la vez permite su utilización desde la Web, esta implementado sobre ASP 5.0 utilizando VB Script y es generado automáticamente por la herramienta Microsoft SOAP Toolkit.
COM Wrapper Application VB 6.0	Componente que encapsula la lógica de transformación y adaptación a formato XML de la respuesta del COM heredado. Realiza la instanciación de un COM, obteniendo el código del servicio invocado de la petición XML y luego ubicándolo en el archivo de configuración ConfigWrapper.xml, es aquí de donde se lee los parámetros de salida y entrada.
COM heredado Application VB 6.0	Es el componente heredado que se reutilizara para manejar parte de la lógica de negocio y el 100% de las operaciones que acceden a las fuentes de información, este componente se encarga de realizar conexiones a una base de datos SQL Server 2000.

Tabla 5.1. Proyectos que componen la arquitectura desarrollada.

Fuente: Creación Propia.

Los archivos de configuración que se utilizan dentro de cada módulo que compone toda la aplicación son los siguientes:

<table>
<tr><td rowspan="2">Aplicación SiteDemo
Detcom (Web Application)</td><td>AppAction.xml</td><td>Contiene las Clases que implementan la lógica de los conectores y las clases intermedias que están asociados a un código de servicio específico.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Actions>
    <Action ID="A001" Class="com.detcom.logistica.complus.ComInvoker" />
    <Action ID="B001" Class="com.detcom.logistica.complus.ComInvoker" />
    <Action ID="B002" Class="com.detcom.logistica.complus.ComInvoker" />
    <Action ID="C001" Class="com.detcom.logistica.complus.ComInvoker" />
    <Action ID="M001" Class="com.detcom.logistica.complus.ComInvoker" />
    <Action ID="BoProv004" Class="com.detcom.logistica.complus.ComInvoker" />
    <Action ID="GenericBo" Class="com.detcom.logistica.Dao.GetDataGeneric" />
</Actions>
```

</td></tr>
<tr><td>ServiceConfig.xml</td><td>Contiene la configuración del Servicio que se invocara, aquí se registran los parámetros de entrada que el servicio necesita tomar de la petición que se encuentra cargada en la sesión del usuario.

```
<trx id="BoProv004">
    <input>
        <data tag="nroalmacen" from="/SvcRq/SvcRqData/almacen" />
    </input>
    <output>
        <data tag="Status" from="/Status/*" />
        <data tag="Data" from="/SvcRsData/*" />
    </output>
</trx>
```

</td></tr>
</table>

<table>
<tr>
<td></td>
<td>WebServiceComplusDef
.xml</td>
<td>En este archivo se configura únicamente la URL de donde se consume el Servicio Web

```
<?xml version="1.0" encoding="iso-8859-1"?>
<Connection>
    <Param>
        <URI>http://localhost/WrapperSOAP/WrapperSOAP.ASP</URI>
    </Param>
</Connection>
```

</td>
</tr>
<tr>
<td>Detcom-bussines
(Java Application)</td>
<td>Module-config.xml</td>
<td>Aquí se encuentra registrada la clase que implementa al conector genérico que trabaja con peticiones/respuestas basadas en XML.

Realiza invocación de servicio BoProv004, y utiliza al conector genérico DetcomAnalyzerImpl que es el que implementa el soporte de intercambio de mensajes XML.

```
<bean id="BoProv004" class="MConnector.HostTransaction" singleton="false">
    <property name="connector">
        <value>com.detcom.analizer.DetcomAnalyzerImpl</value>
    </property>
    <property name="online">
        <value>true</value>
    </property>
    <property name="append">
        <value>false</value>
    </property>
</bean>
```


Realiza invocación de clase intermedia, el resultado de invocar esta clase es un método que devuelve un tipo de dato cadena pero en formato XML ya que el conector genérico que usa es DetcomAnalyzerImpl que es el que implementa el soporte de intercambio de mensajes XML.</td>
</tr>
</table>

<table>
<tr>
<td></td>
<td></td>
<td>
<pre>
<bean id="GenericBo" class="MConnector.HostTransaction" singleton="false">
<property name="connector">
<value>com.detcom.analizer.DetcomAnalyzerImpl</value>
</property>
<property name="online">
<value>true</value>
</property>
<property name="append">
<value>false</value>
</property>
</bean>
</pre>
Los conectores registrados en AppAction.xml deben coincidir con la implementación del conector genérico y además implementar la lógica de la llamada al conector ya que la clase del conector genérico solo implementa métodos de soporte para intercambio y tratamiento de la data en formato XML, el campo de asociación para ambos archivos es el código de servicio.
</td>
</tr>
<tr>
<td>COM Wrapper
Application VB 6.0</td>
<td>ConfigWrapper.xml</td>
<td>
En este archivo se configura los parámetros de entrada y salida que recibe el COM heredado, así como el componente a invocar, la clase y el método que se necesita reutilizar.
<pre>
<Service ID="BoProv004" Component="Detcom.clsCollectionGeneric"
Method="listarInventarioxAlmacen" ResponseAsRecordset="1">
<Input>
<Parameters Type="String" From="/SvcRq/SvcRqData/nroalmacen" />
</Input>
<Output>
<Property Name="propMsgError"/>
</Output>
</Service>
</pre>
</td>
</tr>
</table>

Tabla 5.2. Lista de archivos de configuración XML que se utilizan dentro de cada modulo que componen toda la arquitectura.

Fuente: Creación Propia.

5.5 Sistema heredado

El sistema heredado en su totalidad está formado por un conjunto de interfaces de usuario y componentes COM bien definidos, dichos módulos de software son diferentes e independientes entre sí, donde cada uno brinda una funcionalidad diferente al sistema.

Debido a estas características es factible adaptar el componente que sea necesario e indispensable de reutilizar; para el caso de estudio, el sistema de cuentas por pagar, se requirió solo reutilizar el modulo de control de proveedores y Kardex para el nuevo sistema Web.

El Componente a reutilizar es el componente heredado que se relaciona a dicho modulo y es el que contiene parte de la lógica de negocios y el total de consultas hacia la base de datos.

El Sistema Heredado, que se muestra en esta sección esta desarrollado bajo la plataforma de Visual Basic 6.0 y un servidor de Base de Datos en SQL 2000. La aplicación cumple con las siguientes funcionalidades:

- Registro, actualización y eliminación de órdenes de compra.
- Registro de Insumos o suministros.
- Mantenimiento de Proveedores.
- Mantenimiento de Inventarios.
- Consulta de Kardex.
- Registro de guía de ingreso y de salida.

A continuación se muestra el diagrama de base de datos sobre el cual el COM heredado realiza consultas y transacciones, estas transacciones están relacionadas al modulo de control de proveedores y Kardex.

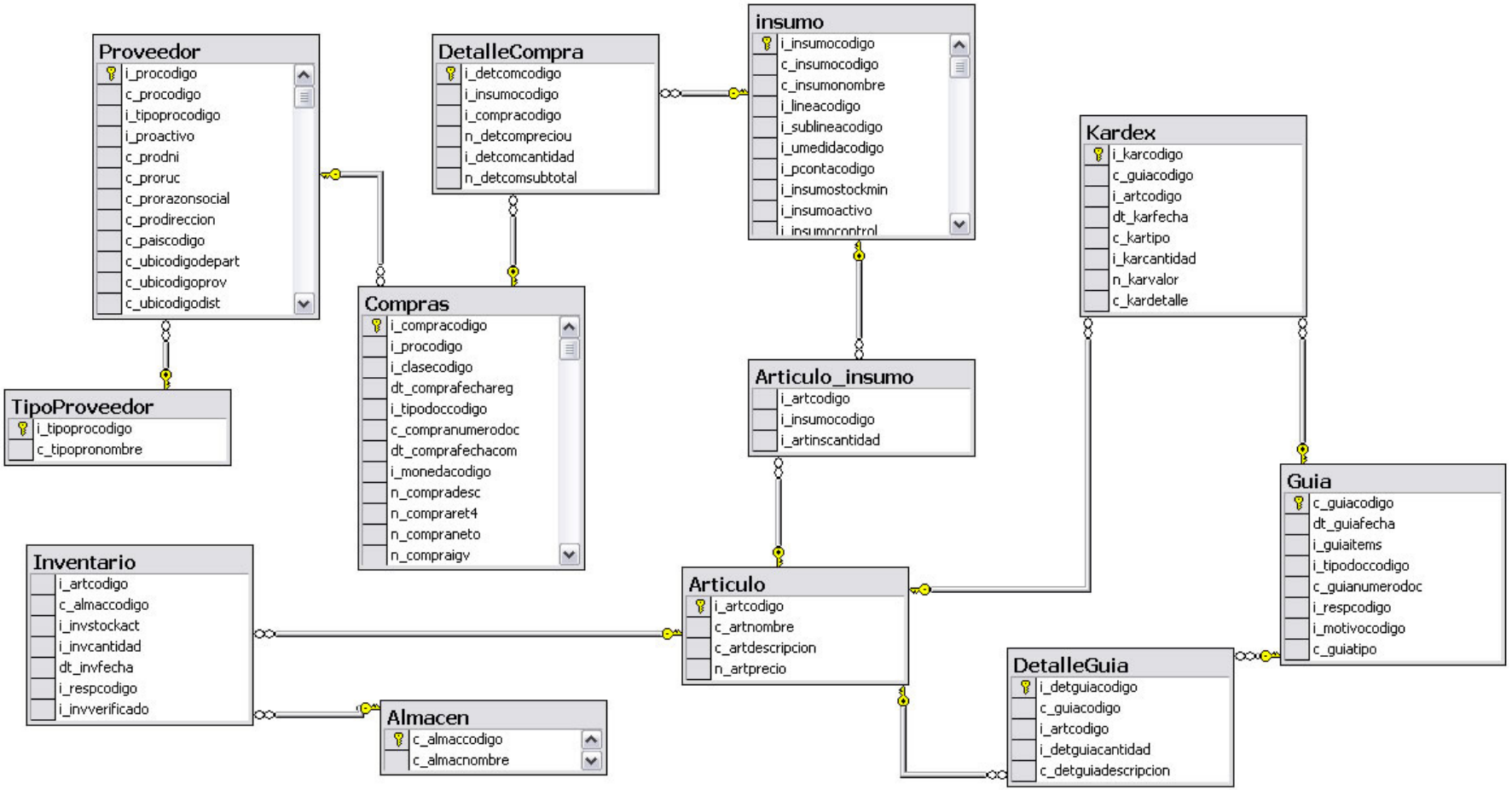


Figura 5.3. Diagrama Entidad Relación del modulo de control de proveedores y Kardex.

Fuente: Creación Propia.

5.6 Web Services Wrapper

El COM Wrapper que es el componente adaptador necesita ser publicado como servicio Web para poder ser consumido por una aplicación que utilice tecnología Web, para lograr esto se utilizo la herramienta Microsoft SOAP Toolkit que mediante una simple configuración logra publicar el componente como un servicio en un repositorio UDDI, de donde las aplicaciones que quieran utilizarlo puedan invocarlo.

Los Servicios Web por ser una tecnología que brinda estándares de comunicación y protocolos de simple uso en Internet, otorgan una infraestructura de comunicación para el intercambio de información entre plataformas de distintas tecnologías, ya que se basan en el uso de interfaces estándares basadas en XML como son los documentos SOAP que constituye el documento XML de transferencia e intercambio de información petición/respuesta entre dos aplicaciones que usan esta tecnología.

A continuación se muestra el segmento de código generado para el Servicio Web que expondrá el COM Wrapper.

```
<%@ LANGUAGE=VBScript %>
<%
Option Explicit
On Error Resume Next
Response.ContentType = "text/xml"
Dim SoapServer
If Not Application("WrapperSOAPInitialized") Then
  Application.Lock
  If Not Application("WrapperSOAPInitialized") Then
    Dim WSDLFilePath
    Dim WSMLFilePath
    WSDLFilePath = Server.MapPath("WrapperSOAP.wsdl")
    WSMLFilePath = Server.MapPath("WrapperSOAP.wsml")
    Set SoapServer = Server.CreateObject("MSSOAP.SoapServer30")
    If Err Then SendFault "Cannot create SoapServer object. " & Err.Description
    SoapServer.Init WSDLFilePath, WSMLFilePath
    If Err Then SendFault "SoapServer.Init failed. " & Err.Description
    Set Application("WrapperSOAPServer") = SoapServer
    Application("WrapperSOAPInitialized") = True
  End If
  Application.UnLock
End If
```

Figura 5.4. Segmento de código del Servicio Web que invoca al COM heredado.

Fuente: Creación Propia.

En la Figura 5.4 se muestran las líneas de código que realizan la llamada del Servicio Web. El archivo WrapperSOAP.wsdl contiene la descripción de los servicios que ofrece el COM Wrapper, el archivo WrapperSOAP.wsml contiene la definición de los parámetros que recibe el COM y el nombre del paquete COM con el que está registrado en el administrador de componentes.

Una vez cargados los archivos de configuración del Web Service se instancia el paquete MSSOAP.SoapServer30, que es el que crea el objeto que se encargara de redireccionar las peticiones hacia el COM Wrapper, logrando así la manipulación de un objeto Wrapper a través del Web Service.

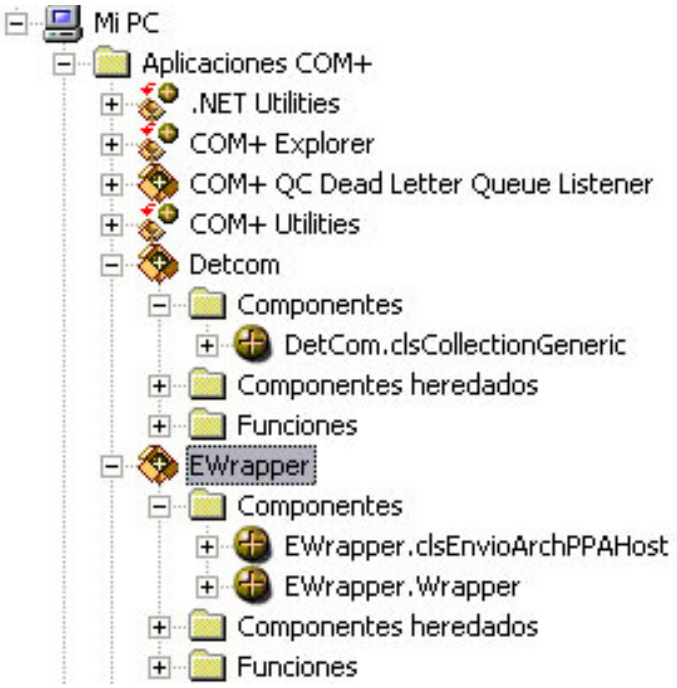


Figura 5.5. Administrador de componentes, COM Wrapper y COM Detcom.
Fuente: Creación Propia.

En la Figura 5.5 se muestra el COM Wrapper registrado en el administrador de componentes, además del COM Detcom que es el componente heredado que se utilizara dentro del desarrollo de la aplicación Web y que es instanciado mediante el COM Wrapper.

En este capítulo hemos explicado detalladamente el funcionamiento de cada elemento o modulo de software que participa dentro del modelo planteado, y como estos interactúan e intercambian información mediante la utilización de las estrategias de integración más aceptadas basándose en la utilización de protocolos

y lenguajes estándares que se vienen usando con más frecuencia en las nuevas tendencias de integración de sistemas.

Capítulo 6: Conclusiones y Recomendaciones

6.1 Conclusiones

- La tendencia de las organizaciones a reutilizar sus antiguos sistemas, surge de la necesidad de implementar sus procesos de negocio, sin tener que incurrir en grandes desarrollos para poder ser utilizados en la construcción de sistemas más modernos basados en las nuevas plataformas tecnológicas; el impulso que ha llevado a las empresas a realizar estas grandes migraciones, ha sido la evolución del análisis de sus procesos negocios y el esfuerzo por lograr una integración horizontal de todas sus áreas organizacionales.

- Las estrategias más exitosas de integración existentes en su mayoría buscan o apuntan a la construcción de adaptadores o interfaces que consigan que distintas aplicaciones intercambien mensajes entre sí utilizando un "lenguaje común" o realizando un formateo o traducción de los mensajes a intercambiar, la cual ha sido y se ha demostrado en el tiempo que es la mejor solución frente a las estrategias que demandan grandes re-desarrollos y por lo tanto cuestan demasiado. Una de las alternativas relacionadas a la construcción de interfaces, ha sido la utilización del lenguaje XML para el intercambio de mensajes, el cual por ser un estándar permite su utilización sin dificultad.

- Una de las razones más importantes por las que se decide la reutilización de sistemas antiguos, es que estos brindan una gran confiabilidad y seguridad, lo cual ha llevado a un gran esfuerzo de poder reutilizarlos sobre las nuevas tendencias en el desarrollo de aplicaciones, esto se puede demostrar con la proliferación del desarrollo de aplicaciones Web sobre Internet ya que estas brindan nuevas capacidades como la distribución, el manejo de lenguajes estándares como el HTML, XML e independencia del sistema operativo etc. Es así como han surgido los Web Services y los Middlewares orientados a mensajes que permiten integrar estos tipos de sistemas antiguos como parte de la funcionalidad de un sistema moderno.

- En nuestra propuesta de trabajo, el lenguaje estándar XML es uno de los elementos más importantes en la construcción de interfaces hacia sistemas heredados. En la solución construida, su utilización se da en todos los niveles de

comunicación de sistemas de distintas tecnologías. Al utilizar Servicios Web se hace uso de documentos XML para el intercambio de mensajes SOAP entre la aplicación Web y el Wrapper, detonando con ello un papel vital en la arquitectura implementada.

6.2 Recomendaciones

- En la medida que seamos capaces de difundir la utilización de estándares ya existentes, se crearan mejores condiciones para la implantación de nuevos métodos de integración.
- Se recomienda el uso de la herramienta propuesta para reducir los tiempos de desarrollo, y economizar recursos, ya que este permite la configuración fácil de los servicios ofrecen los sistemas que se quieren reutilizar.
- Se recomienda la utilización de adaptadores si y solo si los sistemas heredados que se quieren reutilizar manejen tipos de datos estructurados y no simples, a fin de que los Servicios Web que realizaran su integración en la Web, puedan procesarlos.
- Se recomienda la reutilización de los sistemas heredados antes que el redesarrollo, ya que estos son de misión critica y han demostrado en la mayoría de los casos fiabilidad, seguridad y sobretodo estabilidad.

6.3 Trabajos Futuros

Durante el presente estudio se utilizaron dos técnicas de integración de sistemas heredados Wrapping y Web Services, cuyas implementaciones se basan en la utilización de interfaces, y que dan una solución importante en la problemática de integración de sistemas heredados; sin embargo queda como una investigación futura la implementación de Wrappers hacia otros sistemas heredados ya que en esta oportunidad solo se construyo un Wrapper para integración a COM+, una de las investigaciones o estudios futuros seria la construcción de Wrappers hacia programas COBOL, ya que en la mayoría de las instituciones bancarias este lenguaje es muy utilizado y por lo tanto sería muy útil la implementación de esta arquitectura aplicada a ese tipo de sistemas heredados.

ANEXO 1

Herramienta que automatiza la configuración de conectores para la integración de Sistemas Heredados

A continuación se presenta el modelamiento del negocio, diagrama de casos de uso, secuencia y de colaboración, así mismo se detalla las especificaciones de cada proceso del negocio.

A. ALCANCE FUNCIONAL

Requerimientos Funcionales

Nro.	Requerimiento	Descripción
1	**Realizar Mantenimiento de Orden de Compra**	Permite registrar los datos de la Orden de Compra, actualizarlos y eliminarlos.
2	**Realizar Mantenimiento de Suministros**	Permite registrar los productos que estarán disponibles en el almacén, así mismo se podrá realizar actualización y eliminación de los mismos.
3	**Realizar Mantenimiento de Proveedor**	Permite definir los datos generales del proveedor, actualizarlos y eliminarlos.
4	**Gestionar Inventario.**	Permite identificar los ingresos de Inventario.
5	**Consultar Kardex**	Permite consultar todos los movimientos de un determinado producto en un rango de fecha.
7	**Realizar Mantenimiento de Guía de Ingreso**	Permite registrar los datos de la Guía de Ingreso, actualizarlos y eliminarlos.
8	**Realizar Mantenimiento de Guía de Salida**	Permite registrar los datos de la Guía de Salida, actualizarlos y eliminarlos.

Descripción de los procesos de sistema

a) Especificación del Negocio – Realizar Mantenimiento de Orden de Compra

Descripción	Subproceso de **Mantenimiento de Orden de Compra**.
Metas	Registrar la Orden de Compra. Actualizar la Orden de Compra. Eliminar la Orden de Compra.

Flujo de Trabajo	1. Este caso de uso se inicia con la selección de la opción "Mantenimiento de la Orden de Compra" del Menú Principal. 2. El sistema muestra la interfaz de Orden de Compra. 3. Si el actor solicita registrar un Nueva Orden de Compra: 3.1. El sistema habilita la interfaz correspondiente para registrar una orden de compra y solicita el último número para generar el siguiente. Los datos son: tipo de producto, proveedor, solicitud de requerimiento. 3.2. El actor ingresa el tipo de producto. 3.3. El actor selecciona el proveedor. 3.4. El actor selecciona la opción solicitud de requerimiento. 3.5. El sistema muestra la interfaz Buscar solicitud de requerimiento. 3.6. El actor selecciona las solicitudes a ser importadas. 3.7. El actor presiona importar para ingresar las solicitudes a ser requeridas como orden de compra. 3.8. El sistema muestra las solicitudes importadas. 3.9. El actor presiona grabar para grabar la Orden de Compra. 4. El actor tiene la posibilidad de Modificar una Orden de Compra: 4.1. El sistema habilita la interfaz correspondiente para modificar una orden de compra. Los datos pueden ser: proveedor, solicitudes de requerimientos. 4.2. El actor modifica los datos presentados en la interfaz. 4.3. El actor graba los cambios. 4.4. El sistema presenta el listado de la orden de compra modificada. 5. Si el actor solicita Eliminar una Orden de Compra: 5.1. El actor tiene la posibilidad de eliminar o de anular: 5.1.1. Si selecciona eliminar: 5.1.1.1. El actor elimina la orden de compra. 5.1.1.2. El sistema presenta la primera orden de compra. 5.1.2. Si selecciona anular: 5.1.2.1. El actor anula la orden de compra. 6. El sistema presenta la orden de compra anulada.
Flujo Alternativo	No hay flujos alternativos para este caso de uso.

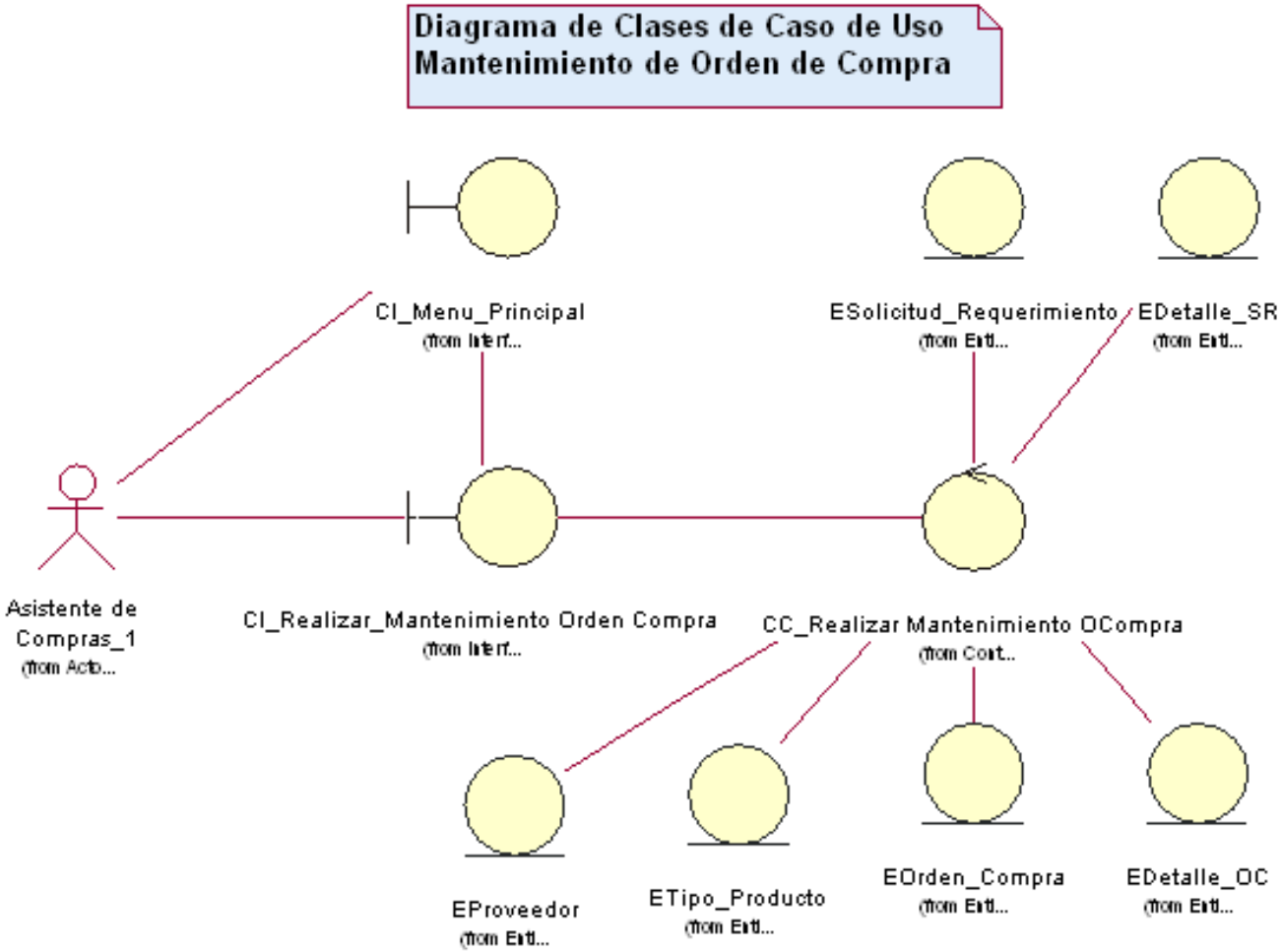
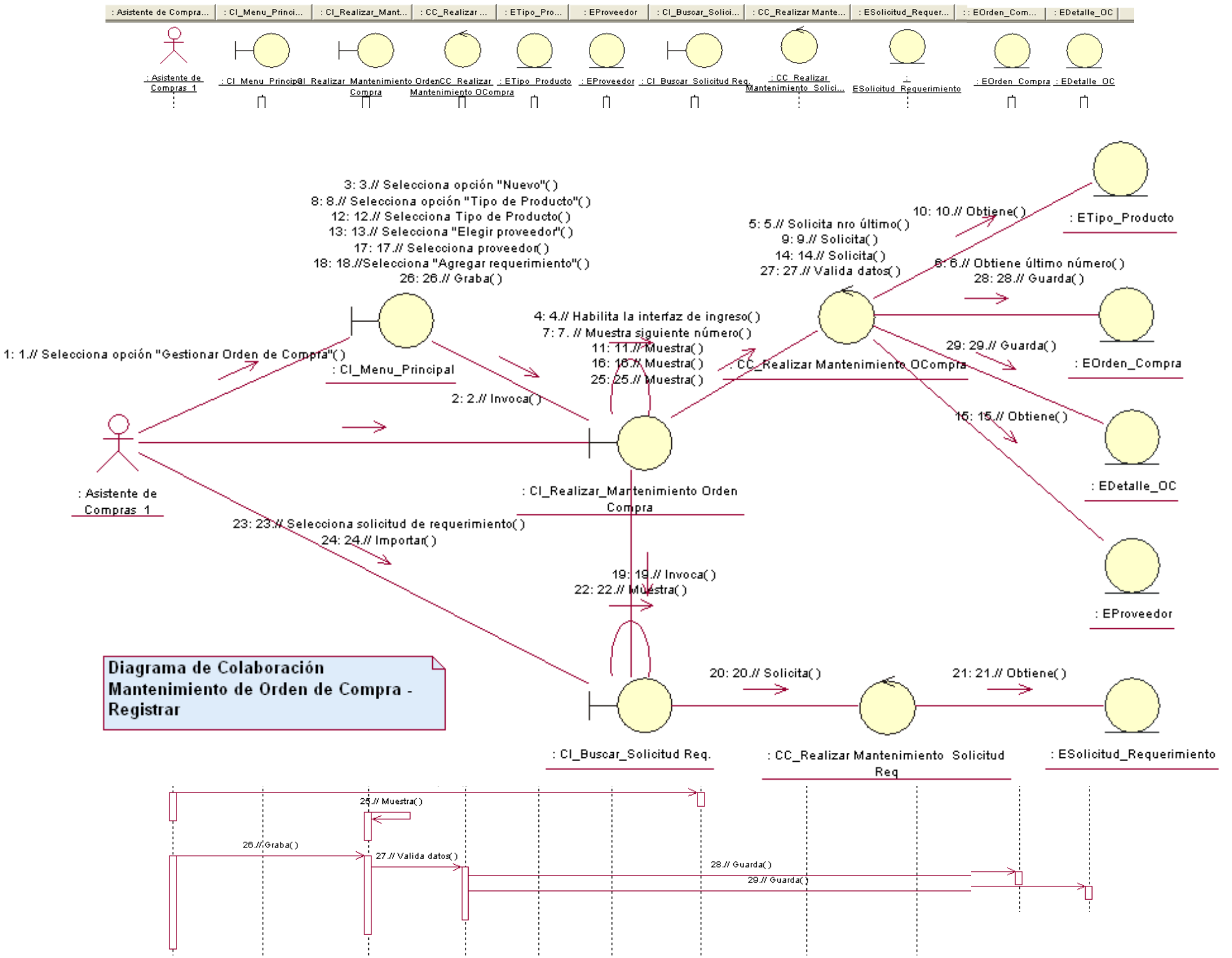
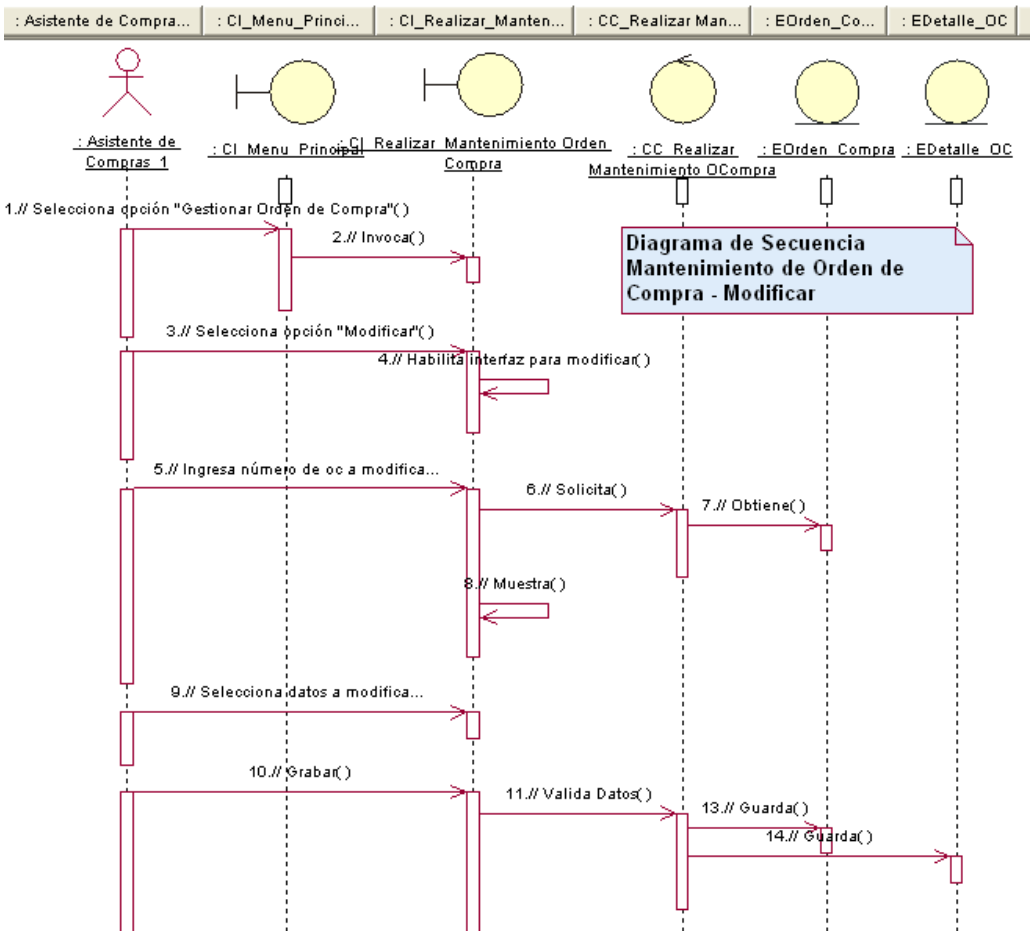


Diagrama de Clases de Caso de Uso
Mantenimiento de Orden de Compra
CI_Menu_Principal
(from Interf...
ESolicitud_Requerimiento
(from Enti...
EDetalle_SR
(from Enti...
Asistente de
Compras_1
(from Acto...
CI_Realizar_Mantenimiento Orden Compra
(from Interf...
CC_Realizar Mantenimiento OCompra
(from Cont...
EProveedor
(from Enti...
ETipo_Producto
(from Enti...
EOrden_Compra
(from Enti...
EDetalle_OC
(from Enti...


: Asistente de Compra...
: CI_Menu_Princi...
: CI_Realizar_Mant...
: CC_Realizar ...
: ETipo_Pro...
: EProveedor
: CI_Buscar_Solici...
: CC_Realizar Mante...
: ESolicitud_Requer...
: : EOrden_Com...
: EDetalle_OC
: Asistente de Compras 1
: CI_Menu_Principal
: CI_Realizar_Mantenimiento Orden Compra
: CC_Realizar Mantenimiento OCompra
: ETipo_Producto
: EProveedor
: CI_Buscar_Solicitud Req.
: CC_Realizar Mantenimiento_Solici...
: ESolicitud_Requerimiento
: EOrden_Compra
: EDetalle_OC
1: 1.// Selecciona opción "Gestionar Orden de Compra"( )
2: 2.// Invoca( )
3: 3.// Selecciona opción "Nuevo"( )
4: 4.// Habilita la interfaz de ingreso( )
5: 5.// Solicita nro último( )
6: 6.// Obtiene último número( )
7: 7. // Muestra siguiente número( )
8: 8.// Selecciona opción "Tipo de Producto"( )
9: 9.// Solicita( )
10: 10.// Obtiene( )
11: 11.// Muestra( )
12: 12.// Selecciona Tipo de Producto( )
13: 13.// Selecciona "Elegir proveedor"( )
14: 14.// Solicita( )
15: 15.// Obtiene( )
16: 16.// Muestra( )
17: 17.// Selecciona proveedor( )
18: 18.//Selecciona "Agregar requerimiento"( )
19: 19.// Invoca( )
20: 20.// Solicita( )
21: 21.// Obtiene( )
22: 22.// Muestra( )
23: 23.// Selecciona solicitud de requerimiento( )
24: 24.// Importar( )
25: 25.// Muestra( )
26: 26.// Graba( )
27: 27.// Valida datos( )
28: 28.// Guarda( )
29: 29.// Guarda( )
Diagrama de Colaboración
Mantenimiento de Orden de Compra - Registrar
: Asistente de Compras 1
: CI_Menu_Principal
: CI_Realizar_Mantenimiento Orden Compra
: CC_Realizar Mantenimiento OCompra
: ETipo_Producto
: EProveedor
: CI_Buscar_Solicitud Req.
: CC_Realizar Mantenimiento Solicitud Req
: ESolicitud_Requerimiento
: EOrden_Compra
: EDetalle_OC
25.// Muestra( )
26.// Graba( )
27.// Valida datos( )
28.// Guarda( )
29.// Guarda( )


: Asistente de Compra...
: CI_Menu_Princi...
: CI_Realizar_Manten...
: CC_Realizar Man...
: EOrden_Co...
: EDetalle_OC
: Asistente de Compras 1
: CI Menu Principal
: CI Realizar Mantenimiento Orden Compra
: CC Realizar Mantenimiento OCompra
: EOrden Compra
: EDetalle OC
1.// Selecciona opción "Gestionar Orden de Compra"( )
2.// Invoca( )
Diagrama de Secuencia Mantenimiento de Orden de Compra - Modificar
3.// Selecciona opción "Modificar"( )
4.// Habilita interfaz para modificar( )
5.// Ingresa número de oc a modifica...
6.// Solicita( )
7.// Obtiene( )
8.// Muestra( )
9.// Selecciona datos a modifica...
10.// Grabar( )
11.// Valida Datos( )
13.// Guarda( )
14.// Guarda( )

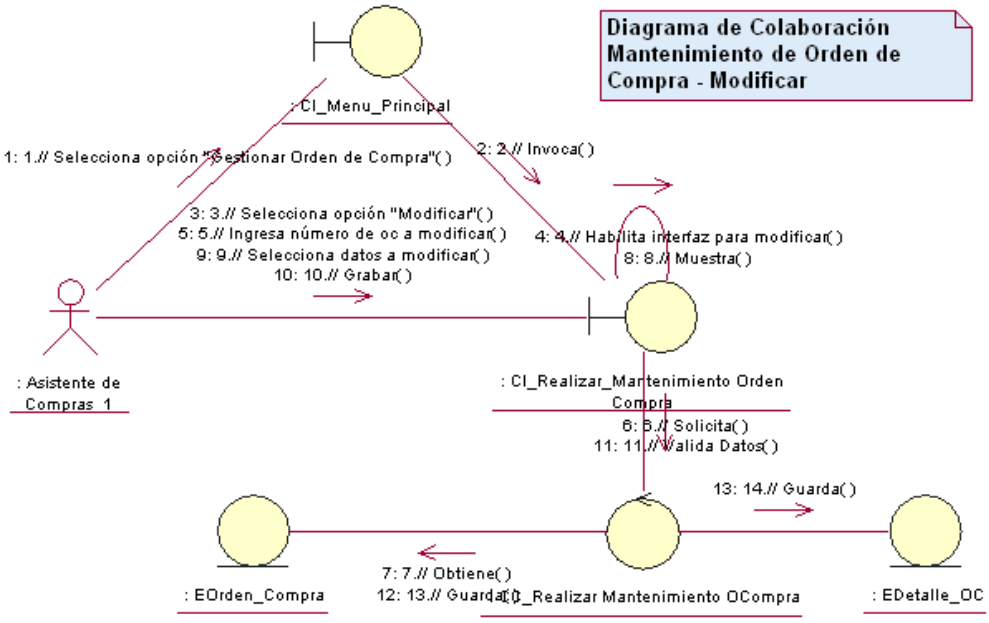
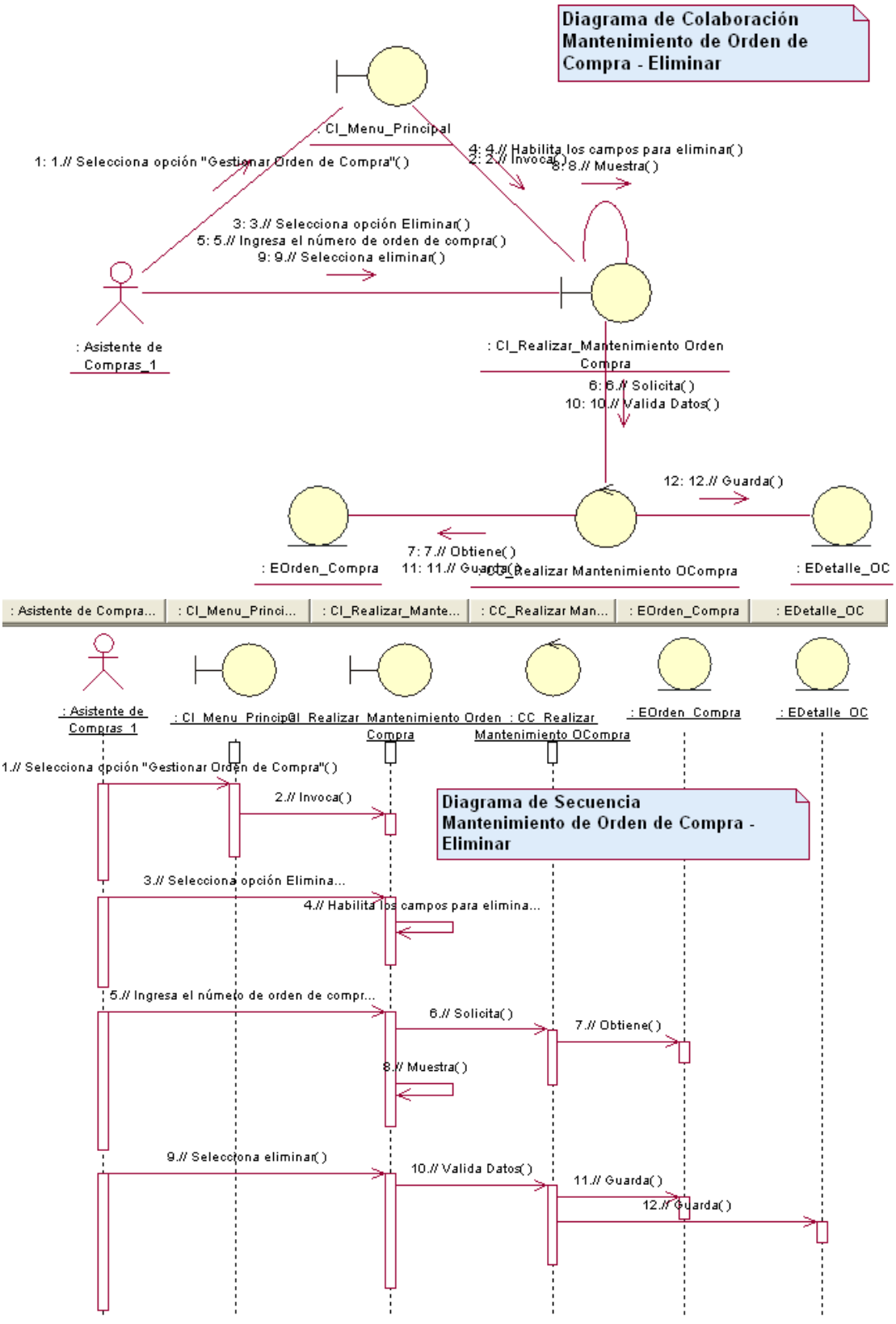


Diagrama de Colaboración Mantenimiento de Orden de Compra - Modificar
: CI_Menu_Principal
1: 1.// Selecciona opción "Gestionar Orden de Compra"( )
2: 2.// Invoca( )
3: 3.// Selecciona opción "Modificar"( )
5: 5.// Ingresa número de oc a modificar( )
9: 9.// Selecciona datos a modificar( )
10: 10.// Grabar( )
4: 4.// Habilita interfaz para modificar( )
8: 8.// Muestra( )
: Asistente de Compras 1
: CI_Realizar_Mantenimiento Orden Compra
6: 6.// Solicita( )
11: 11.// Valida Datos( )
13: 14.// Guarda( )
7: 7.// Obtiene( )
12: 13.// Guarda( )
: EOrden_Compra
: CC_Realizar Mantenimiento OCompra
: EDetalle_OC



Descripción de los procesos de sistema

b) Especificación del Negocio – Realizar Mantenimiento de Proveedor

Descripción	Subproceso de **Realizar Mantenimiento de Proveedor**.
Metas	Registrar datos de proveedor. Eliminar datos de proveedor. Actualizar datos de proveedor.
Flujo de Trabajo	1. Este caso de uso se inicia con la selección de la opción "Mantenimiento de Proveedor" del Menú Principal. 2. El sistema muestra la interfaz de Mantenimiento de Proveedor que contiene una lista de proveedores con las opciones para registrar, modificar, eliminar o salir. 3. Si el actor solicita registrar un Nuevo Proveedor: 3.1. El sistema habilita la interfaz correspondiente para registrar un proveedor. Los datos son: código, activo, ruc, dni, razón social, tipo,

	dirección, país, departamento, provincia, distrito, teléfonos, faxes, giro del negocio, e mail, contactos, bancos. 3.2. El actor ingresa los datos: código, activo, ruc, dni, razón social, teléfonos, faxes, mail, dirección. 3.3. El actor selecciona elegir tipo de proveedor (PT/ insumos /útiles) de una lista de proveedores [Giro del negocio]. 3.4. El actor selecciona el tipo de proveedor. 3.5. El actor selecciona elegir país de una lista de países. 3.6. El actor selecciona el país. 3.7. El actor selecciona elegir departamento de una lista de departamentos. 3.8. El actor selecciona el departamento. 3.9. El actor selecciona elegir provincia de una lista de provincias. 3.10. El actor selecciona la provincia. 3.11. El actor selecciona elegir distrito. 3.12. El actor selecciona el distrito. 3.13. El actor selecciona elegir banco. 3.14. El actor selecciona el banco. 3.15. El actor graba. 3.16. El sistema presenta el listado de proveedores. 4. El actor tiene la posibilidad de Modificar un Proveedor: 4.1. El sistema habilita la interfaz correspondiente para modificar un proveedor. Los datos pueden ser : activo, ruc, dni, razón social, tipo, dirección, país, departamento, provincia, distrito, teléfonos, faxes, giro del negocio, e mail, contactos, bancos. 4.2. El actor modifica los datos presentados en la interfaz. 4.3. El actor graba los cambios. 4.4. El sistema presenta el listado de proveedores. 5. Si el actor solicita Eliminar un Proveedor: 5.1. El actor elimina un proveedor. 5.2. El sistema presenta el listado de proveedores.
Flujo Alternativo	No hay flujos alternativos para este caso de uso.

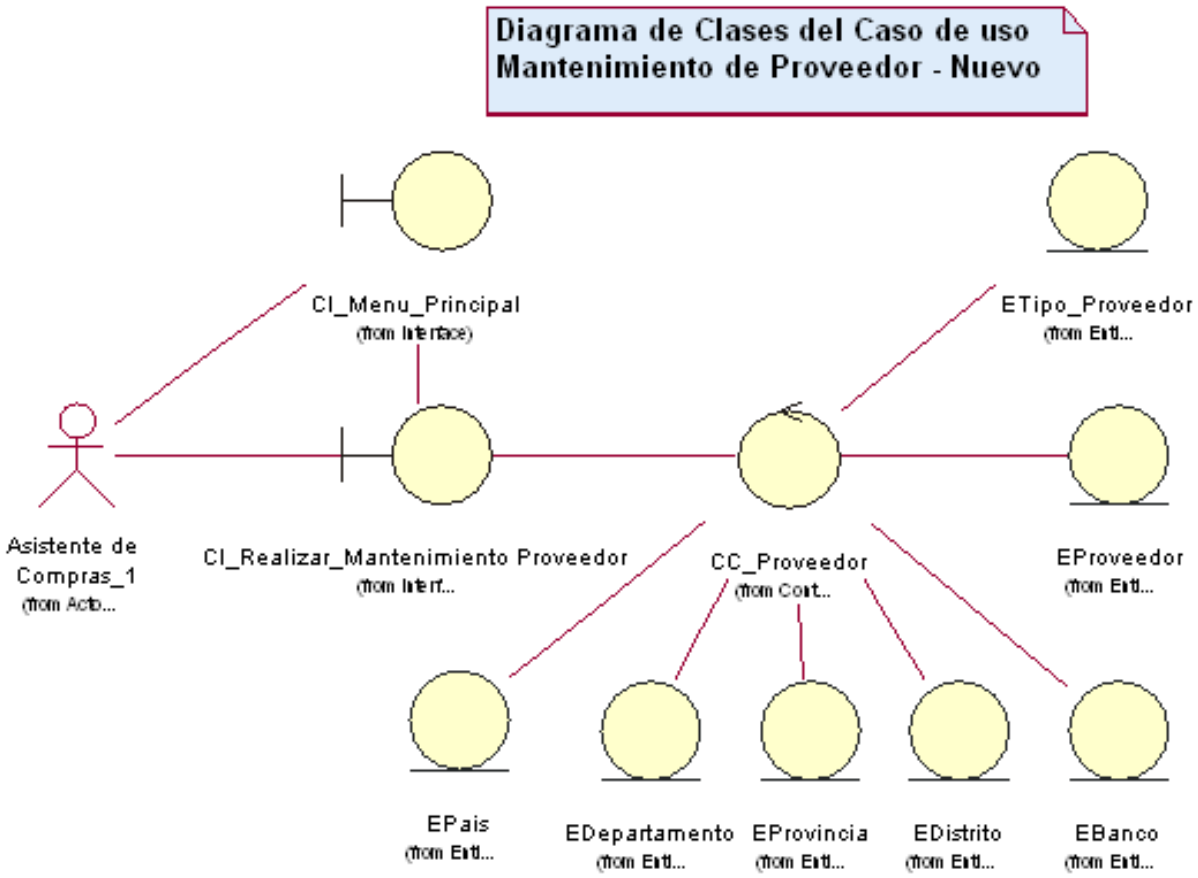
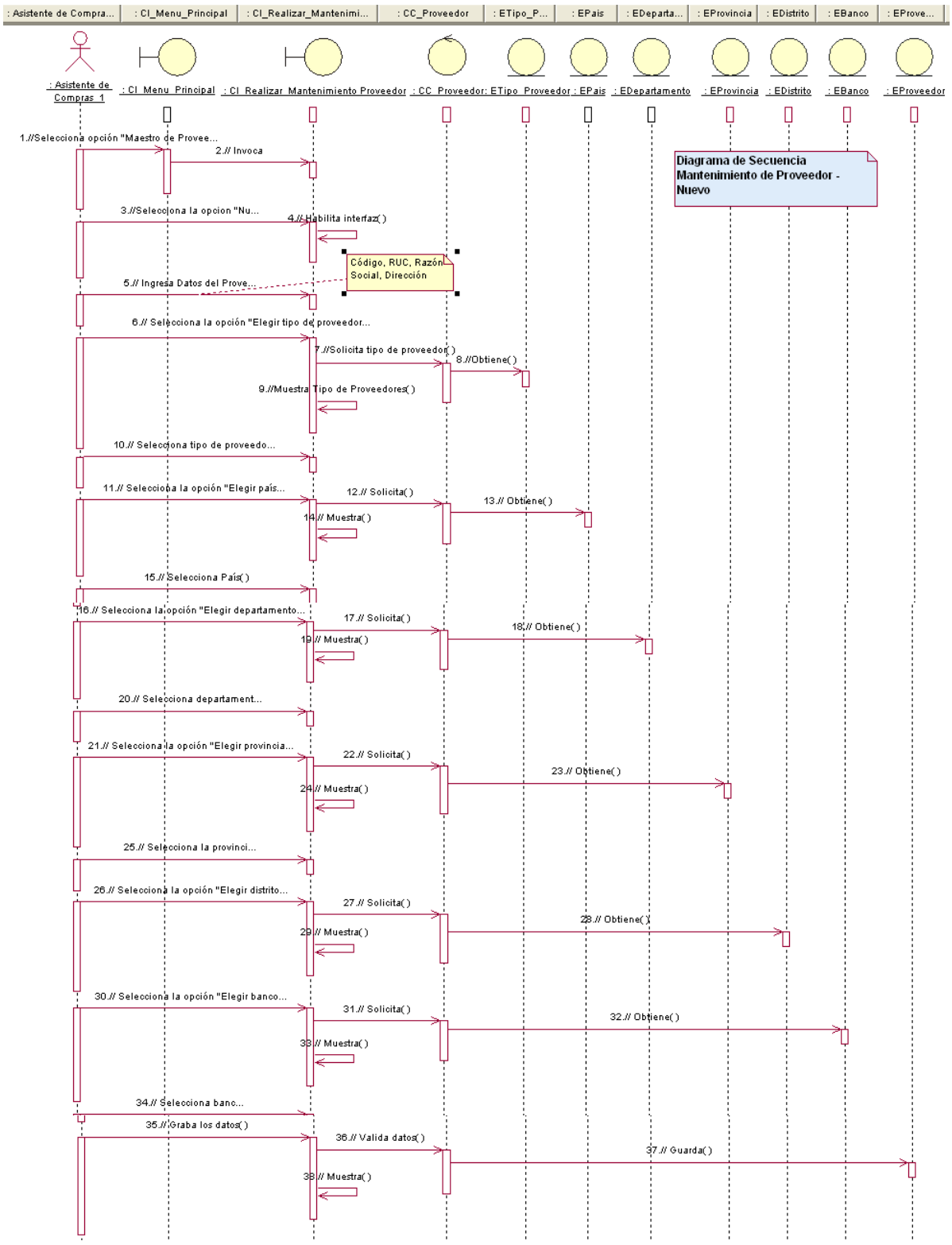


Diagrama de Clases del Caso de uso
Mantenimiento de Proveedor - Nuevo
CI_Menu_Principal
(from Interface)
ETipo_Proveedor
(from Enti...
Asistente de
Compras_1
(from Acto...
CI_Realizar_Mantenimiento Proveedor
(from Interf...
CC_Proveedor
(from Cont...
EProveedor
(from Enti...
EPais
(from Enti...
EDepartamento
(from Enti...
EProvincia
(from Enti...
EDistrito
(from Enti...
EBanco
(from Enti...


: Asistente de Compra...
: CI_Menu_Principal
: CI_Realizar_Mantenimi...
: CC_Proveedor
: ETipo_P...
: EPais
: EDeparta...
: EProvincia
: EDistrito
: EBanco
: EProve...
: Asistente de Compras 1
: CI_Menu_Principal
: CI_Realizar_Mantenimiento Proveedor
: CC_Proveedor
: ETipo_Proveedor
: EPais
: EDepartamento
: EProvincia
: EDistrito
: EBanco
: EProveedor
Diagrama de Secuencia Mantenimiento de Proveedor - Nuevo
1.//Selecciona opción "Maestro de Provee...
2.// Invoca
3.//Selecciona la opcion "Nu...
4.//Habilita interfaz()
Código, RUC, Razón Social, Dirección
5.// Ingresa Datos del Prove...
6.// Selecciona la opción "Elegir tipo de proveedor...
7.//Solicita tipo de proveedor()
8.//Obtiene()
9.//Muestra Tipo de Proveedores()
10.// Selecciona tipo de proveedo...
11.// Selecciona la opción "Elegir país...
12.// Solicita()
13.// Obtiene()
14.// Muestra()
15.// Selecciona País()
16.// Selecciona la opción "Elegir departamento...
17.// Solicita()
18.// Obtiene()
19.// Muestra()
20.// Selecciona departament...
21.// Selecciona la opción "Elegir provincia...
22.// Solicita()
23.// Obtiene()
24.// Muestra()
25.// Selecciona la provinci...
26.// Selecciona la opción "Elegir distrito...
27.// Solicita()
28.// Obtiene()
29.// Muestra()
30.// Selecciona la opción "Elegir banco...
31.// Solicita()
32.// Obtiene()
33.// Muestra()
34.// Selecciona banc...
35.// Graba los datos()
36.// Valida datos()
37.// Guarda()
38.// Muestra()

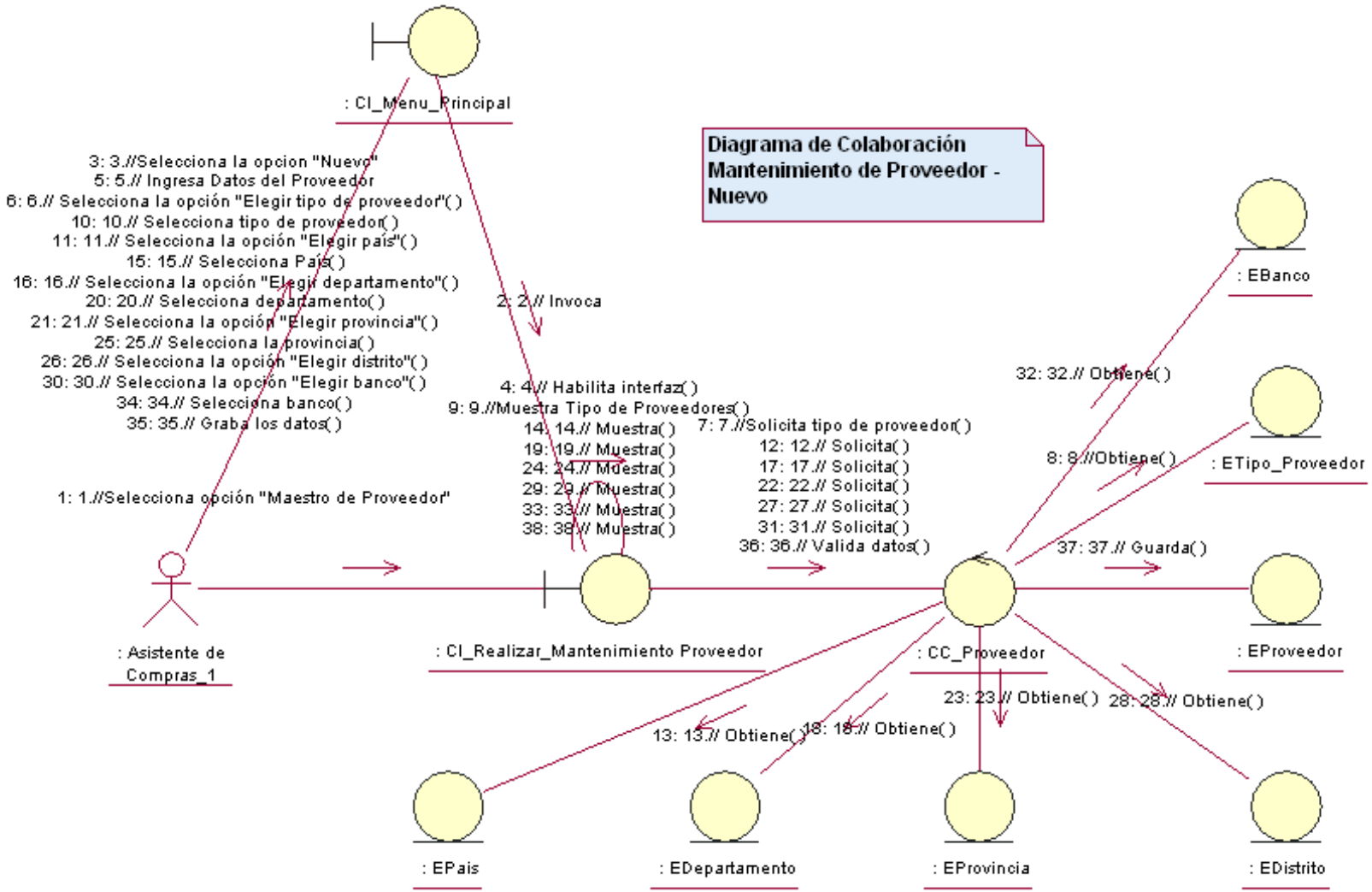
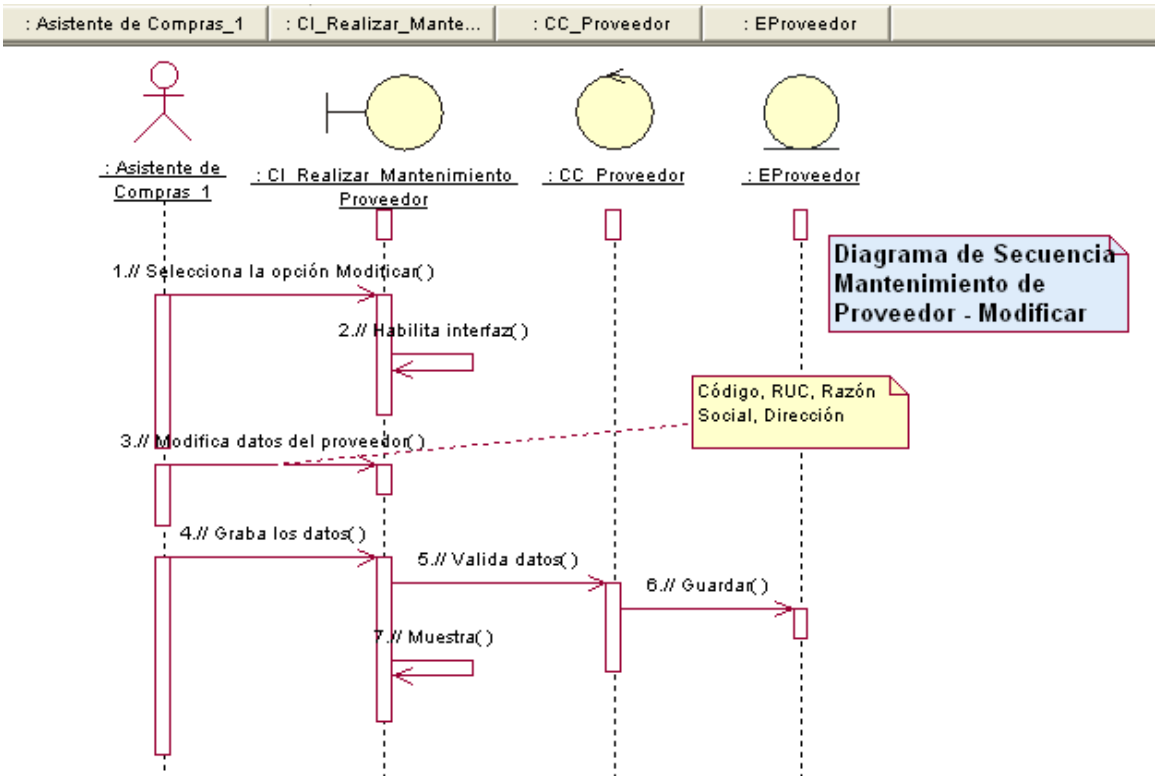
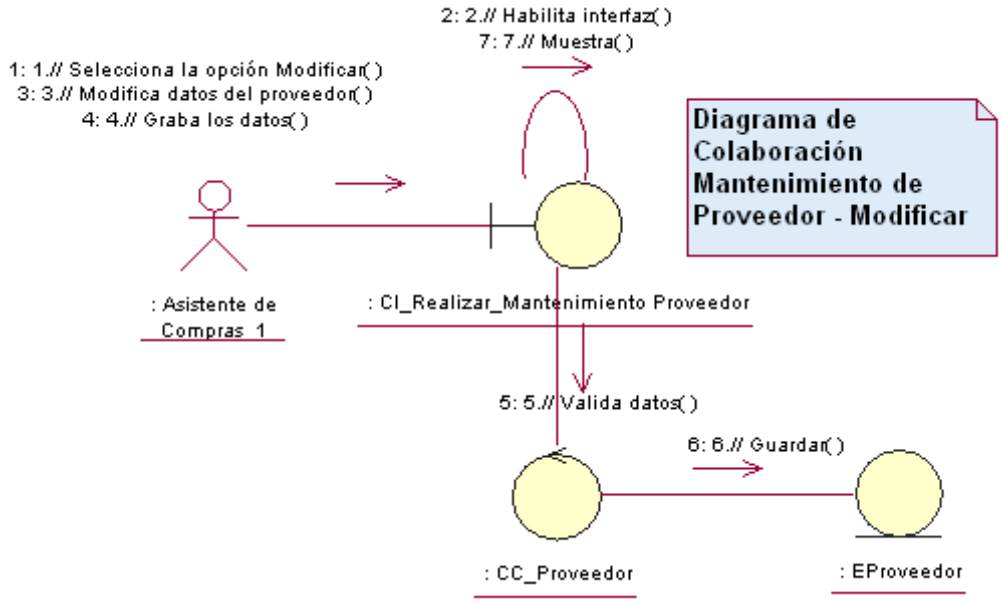
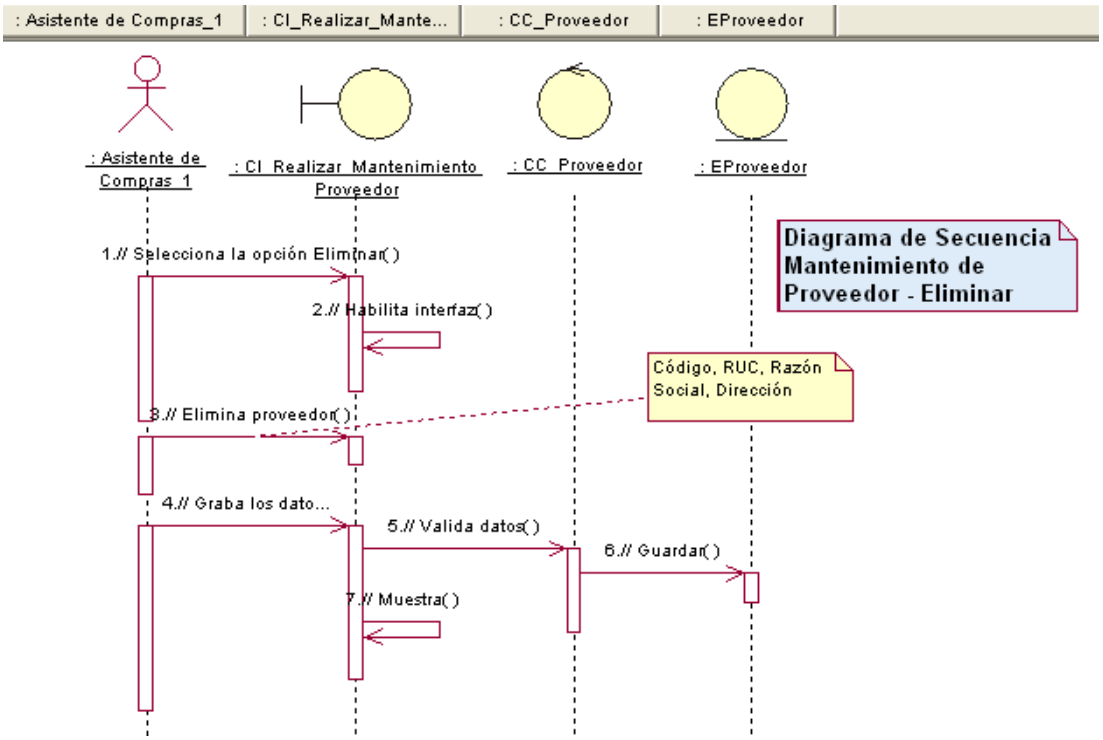
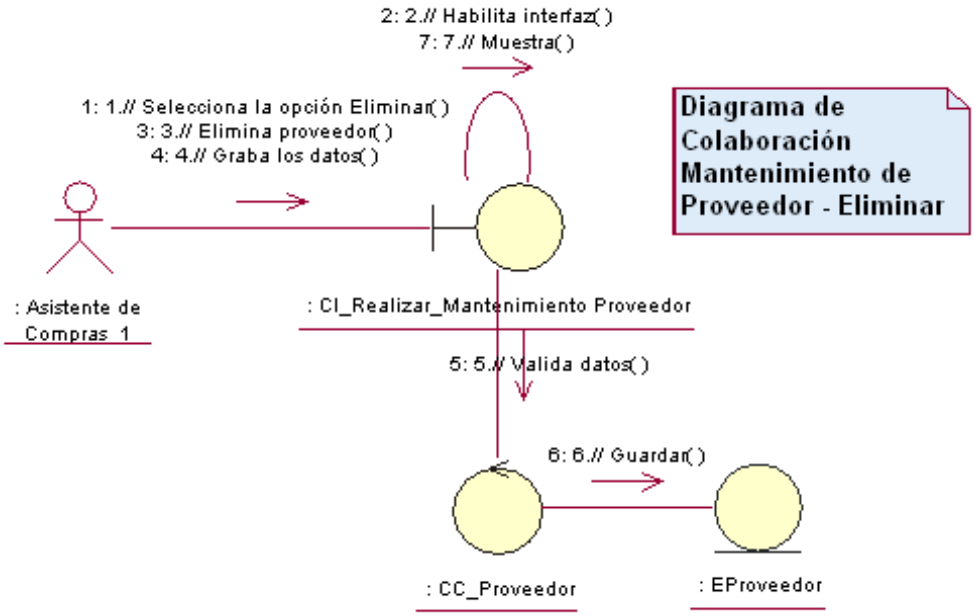


Diagrama de Colaboración
Mantenimiento de Proveedor -
Nuevo
: CI_Menu_Principal
: Asistente de
Compras_1
: CI_Realizar_Mantenimiento Proveedor
: CC_Proveedor
: EBanco
: ETipo_Proveedor
: EProveedor
: EPais
: EDepartamento
: EProvincia
: EDistrito
1: 1.//Selecciona opción "Maestro de Proveedor"
2: 2.// Invoca
3: 3.//Selecciona la opcion "Nuevo"
4: 4.// Habilita interfaz( )
5: 5.// Ingresa Datos del Proveedor
6: 6.// Selecciona la opción "Elegir tipo de proveedor"( )
7: 7.//Solicita tipo de proveedor( )
8: 8.//Obtiene( )
9: 9.//Muestra Tipo de Proveedores( )
10: 10.// Selecciona tipo de proveedor( )
11: 11.// Selecciona la opción "Elegir país"( )
12: 12.// Solicita( )
13: 13.// Obtiene( )
14: 14.// Muestra( )
15: 15.// Selecciona País( )
16: 16.// Selecciona la opción "Elegir departamento"( )
17: 17.// Solicita( )
18: 18.// Obtiene( )
19: 19.// Muestra( )
20: 20.// Selecciona departamento( )
21: 21.// Selecciona la opción "Elegir provincia"( )
22: 22.// Solicita( )
23: 23.// Obtiene( )
24: 24.// Muestra( )
25: 25.// Selecciona la provincia( )
26: 26.// Selecciona la opción "Elegir distrito"( )
27: 27.// Solicita( )
28: 28.// Obtiene( )
29: 29.// Muestra( )
30: 30.// Selecciona la opción "Elegir banco"( )
31: 31.// Solicita( )
32: 32.// Obtiene( )
33: 33.// Muestra( )
34: 34.// Selecciona banco( )
35: 35.// Graba los datos( )
36: 36.// Valida datos( )
37: 37.// Guarda( )
38: 38.// Muestra( )


: Asistente de Compras_1
: CI_Realizar_Mante...
: CC_Proveedor
: EProveedor
: Asistente de Compras_1
: CI_Realizar_Mantenimiento Proveedor
: CC_Proveedor
: EProveedor
Diagrama de Secuencia Mantenimiento de Proveedor - Modificar
1.// Selecciona la opción Modificar( )
2.// Habilita interfaz( )
Código, RUC, Razón Social, Dirección
3.// Modifica datos del proveedor( )
4.// Graba los datos( )
5.// Valida datos( )
6.// Guardar( )
7.// Muestra( )


2: 2.// Habilita interfaz( )
7: 7.// Muestra( )
1: 1.// Selecciona la opción Modificar( )
3: 3.// Modifica datos del proveedor( )
4: 4.// Graba los datos( )
Diagrama de Colaboración Mantenimiento de Proveedor - Modificar
: Asistente de Compras_1
: CI_Realizar_Mantenimiento Proveedor
5: 5.// Valida datos( )
6: 6.// Guardar( )
: CC_Proveedor
: EProveedor

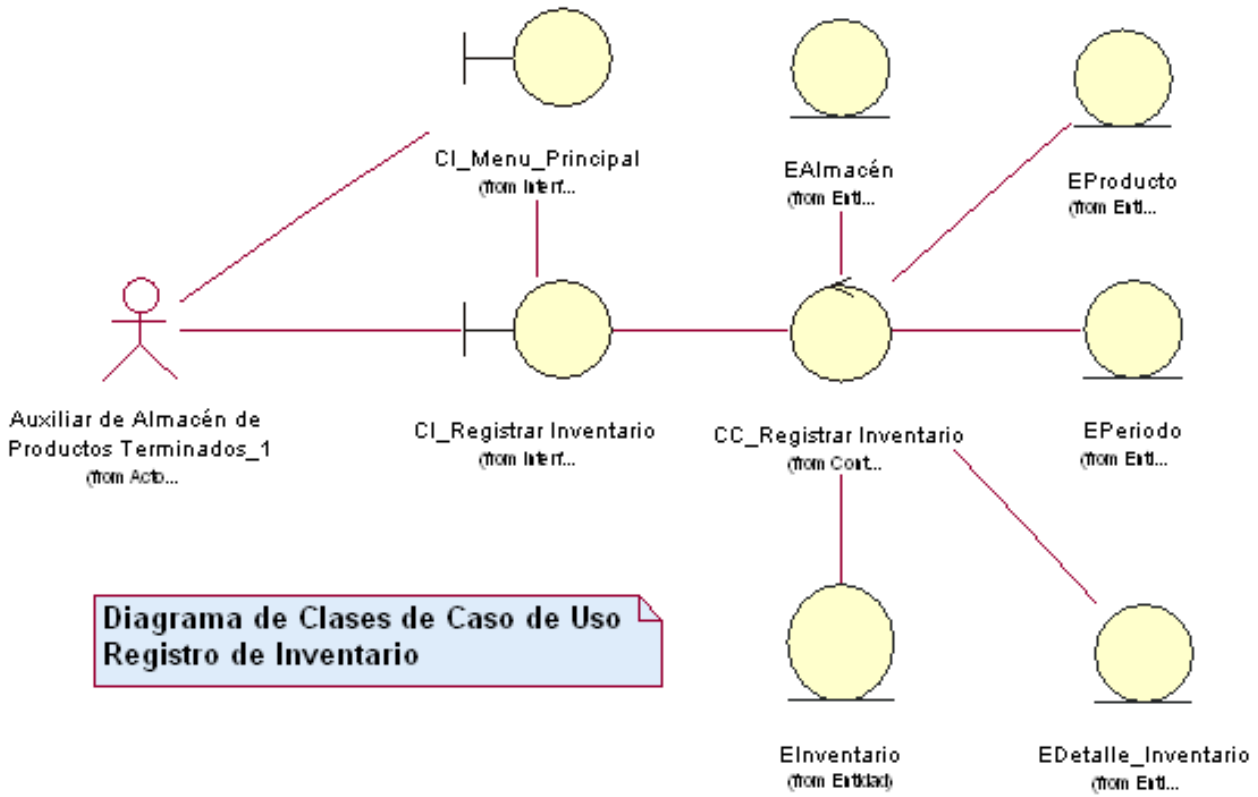

: Asistente de Compras_1
: CI_Realizar_Mante...
: CC_Proveedor
: EProveedor
: Asistente de Compras_1
: CI_Realizar_Mantenimiento Proveedor
: CC_Proveedor
: EProveedor
1.// Selecciona la opción Eliminar( )
2.// Habilita interfaz( )
3.// Elimina proveedor( )
4.// Graba los dato...
5.// Valida datos( )
6.// Guardar( )
7.// Muestra( )
Código, RUC, Razón Social, Dirección
Diagrama de Secuencia Mantenimiento de Proveedor - Eliminar

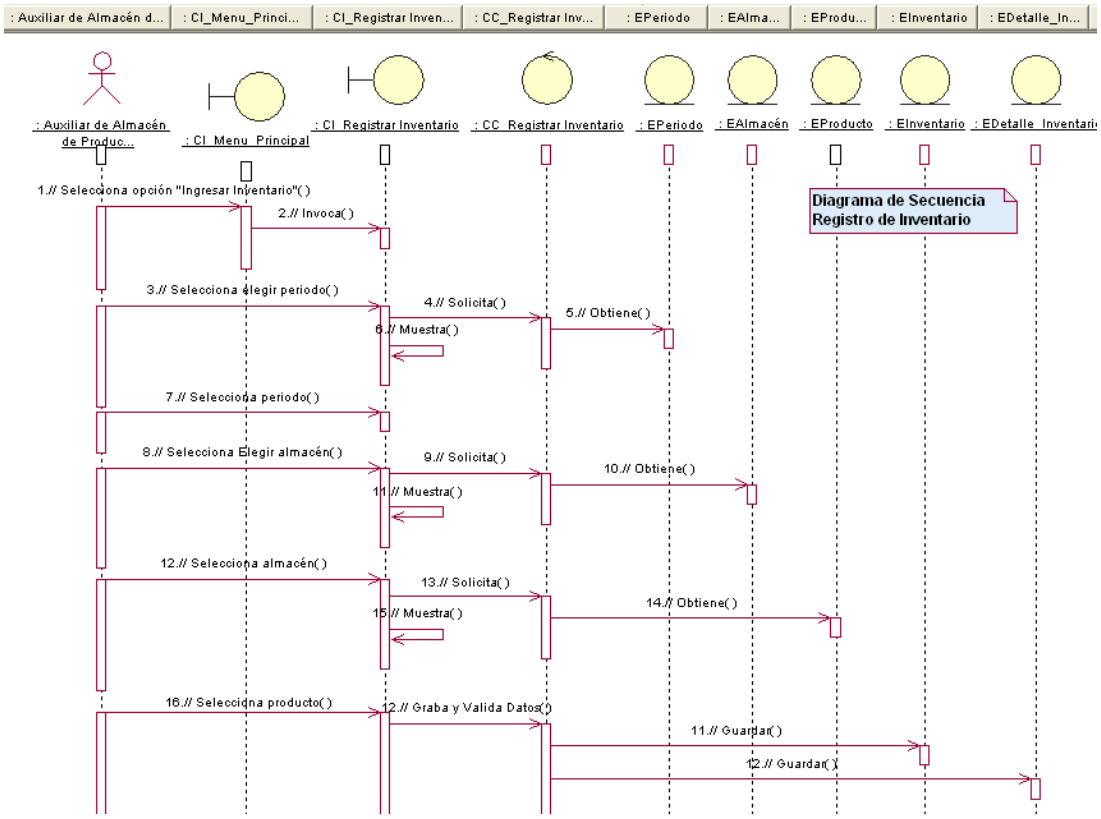

2: 2.// Habilita interfaz( )
7: 7.// Muestra( )
1: 1.// Selecciona la opción Eliminar( )
3: 3.// Elimina proveedor( )
4: 4.// Graba los datos( )
: Asistente de Compras_1
: CI_Realizar_Mantenimiento Proveedor
5: 5.// Valida datos( )
6: 6.// Guardar( )
: CC_Proveedor
: EProveedor
Diagrama de Colaboración Mantenimiento de Proveedor - Eliminar

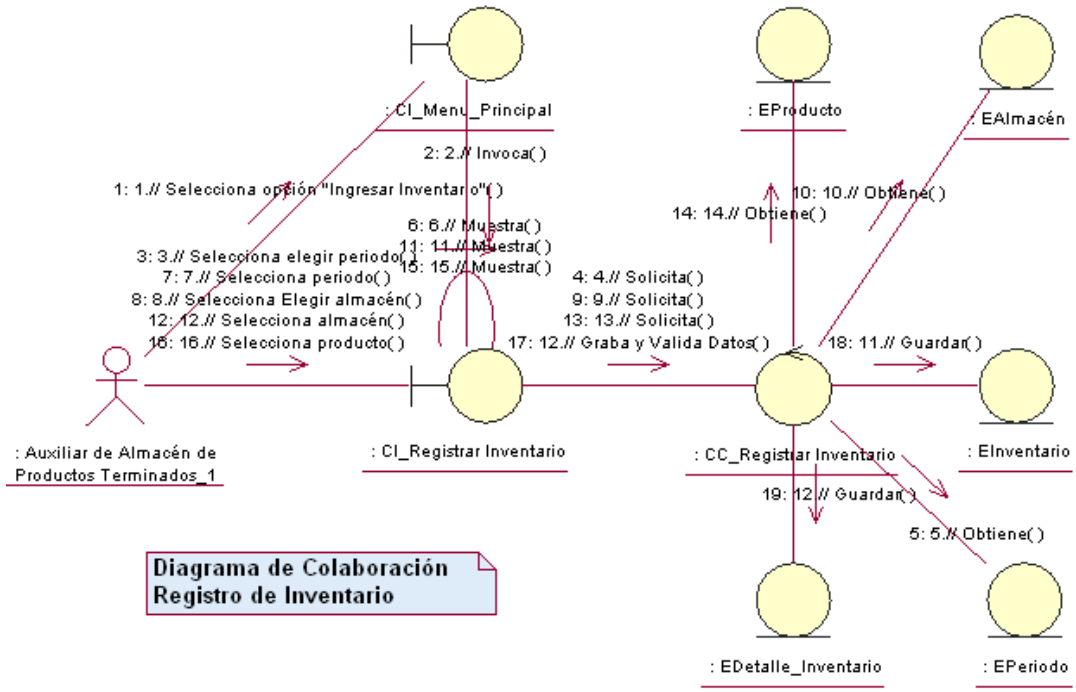
Descripción de los procesos de sistema

c) Especificación del Negocio – Manejar Inventario

Descripción	Subproceso de **Registro de Inventario**.
Metas	Registrar de Inventario.
Flujo de Trabajo	1. Este caso de uso se inicia con la selección de la opción "Recepción de Pedidos en Almacén General" del Menú Principal. 2. El sistema muestra la interfaz de Recepción de Pedidos que contiene el nombre del actor, los pedidos pendientes y su detalle. 3. El actor ingresa el almacén en el que está trabajando. 4. El sistema muestra una lista de almacenes. 5. El actor selecciona una de los almacenes. 6. El actor selecciona con un check los pedidos. 7. El actor graba. 8. El sistema genera por cada uno de los pedidos sus almacenes temporales correspondientes.
Flujo Alternativo	No hay flujos alternativos para este caso de uso.


CI_Menu_Principal
(from Interf...
EAlmacén
(from Enti...
EProducto
(from Enti...
Auxiliar de Almacén de Productos Terminados_1
(from Acto...
CI_Registrar Inventario
(from Interf...
CC_Registrar Inventario
(from Cont...
EPeriodo
(from Enti...
Diagrama de Clases de Caso de Uso Registro de Inventario
EInventario
(from Entidad)
EDetalle_Inventario
(from Enti...

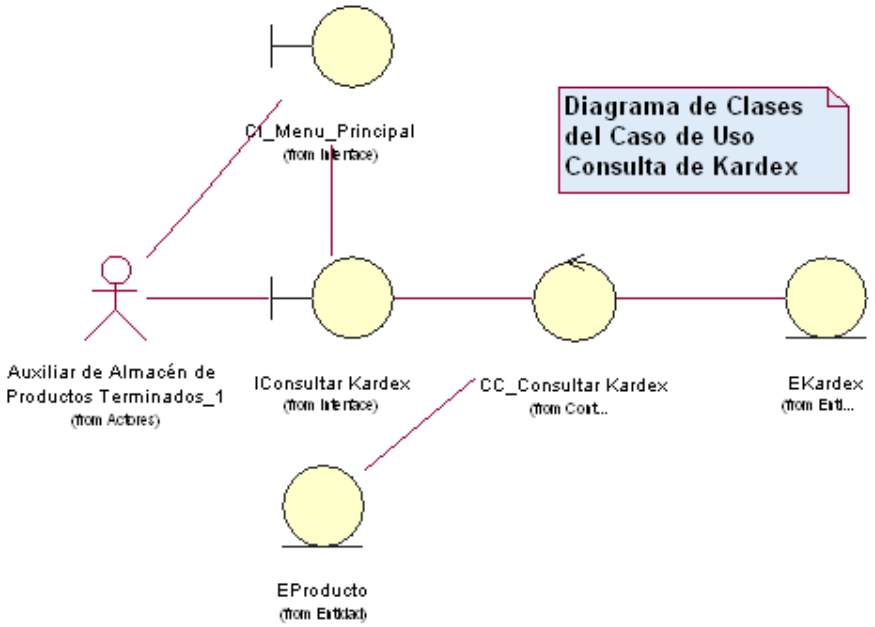


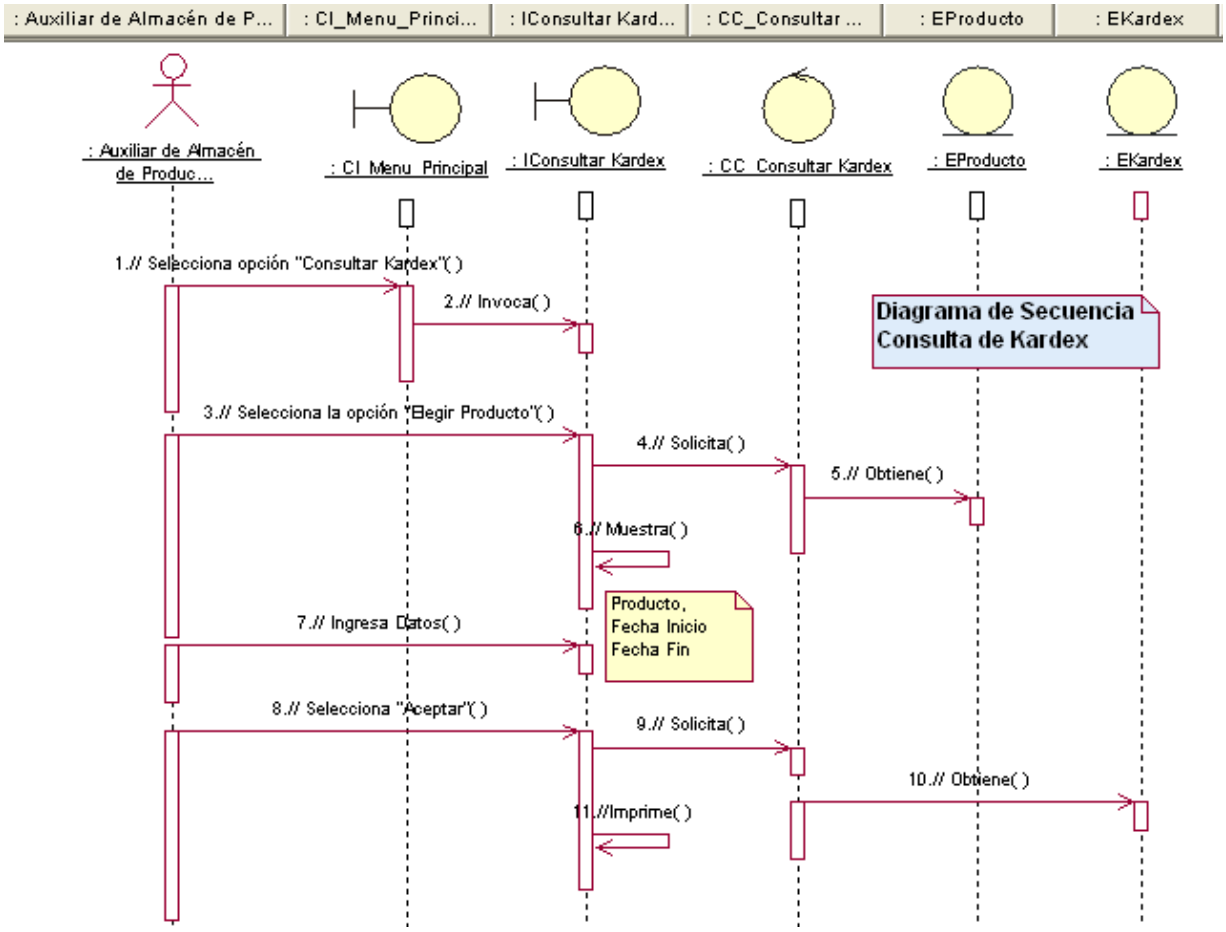


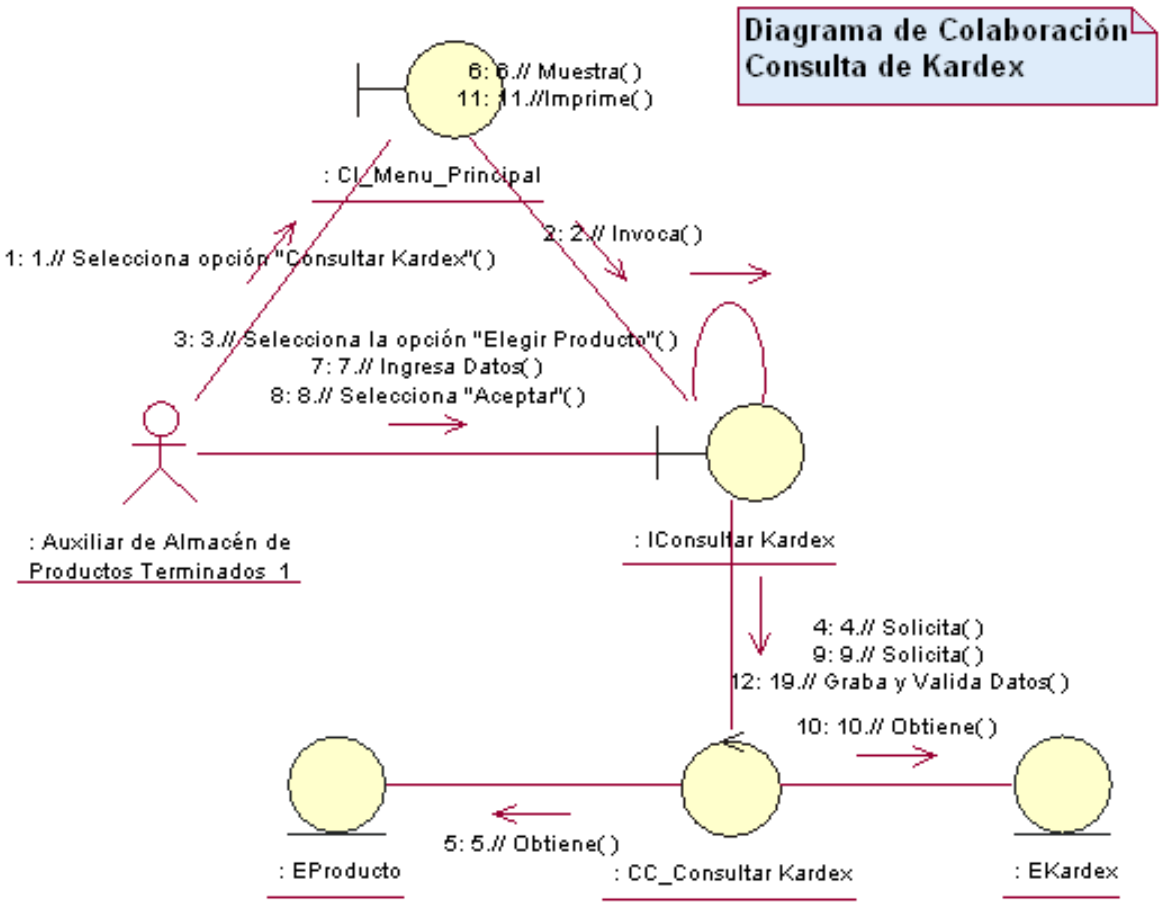
Descripción de los procesos de sistema

d) Especificación del Negocio – Consultar Kardex

Descripción	Subproceso de **Consulta de Kardex**.
Metas	Consultar todos los movimientos de un determinado producto en un rango de fecha.
Flujo de Trabajo	1. Este caso de uso se inicia con la selección de la opción "Consulta de Kardex" del Menú Principal. 2. El sistema muestra la interfaz de Consulta de Kardex que contiene una opción de seleccionar el producto donde aparece una lista de productos a seleccionar, también aparecen fechas de inicio y fin. 3. El actor ingresa los datos (código del producto, fecha de inicio, fecha de fin). 4. El actor selecciona: 4.1. La opción imprimir: Si desea enviarlo a impresión. 4.2. La opción pantalla: Si lo desea visualizar en la pantalla. 4.3. La opción salir: Si desea salir.
Flujo Alternativo	No hay flujos alternativos para este caso de uso.


CI_Menu_Principal
(from Interface)
Diagrama de Clases
del Caso de Uso
Consulta de Kardex
Auxiliar de Almacén de
Productos Terminados_1
(from Actores)
IConsultar Kardex
(from Interface)
CC_Consultar Kardex
(from Cont...
EKardex
(from Enti...
EProducto
(from Entidad)


: Auxiliar de Almacén de P...
: CI_Menu_Princi...
: IConsultar Kard...
: CC_Consultar ...
: EProducto
: EKardex
: Auxiliar de Almacén de Produc...
: CI_Menu_Principal
: IConsultar Kardex
: CC_Consultar Kardex
: EProducto
: EKardex
1.// Selecciona opción "Consultar Kardex"( )
2.// Invoca( )
Diagrama de Secuencia
Consulta de Kardex
3.// Selecciona la opción "Elegir Producto"( )
4.// Solicita( )
5.// Obtiene( )
6.// Muestra( )
Producto,
Fecha Inicio
Fecha Fin
7.// Ingresa Datos( )
8.// Selecciona "Aceptar"( )
9.// Solicita( )
10.// Obtiene( )
11.//Imprime( )



Descripción de los procesos de sistema

e) Especificación del Negocio – Realizar Mantenimiento de Guía de Ingreso

Descripción	Subproceso de **Mantenimiento de Guía de Ingreso**.
Metas	Registrar la Guía de Ingreso. Actualizar la Guía de Ingreso. Eliminar la Guía de Ingreso.

Flujo de Trabajo	1. Este caso de uso se inicia con la selección de la opción "Mantenimiento de la Guía de Ingreso" del Menú Principal. 2. El sistema muestra la interfaz de Guía de Ingreso. 3. Si el actor solicita registrar un Nuevo Guía de Ingreso: 3.1. El sistema habilita la interfaz correspondiente para registrar una guía de ingreso y solicita el último número para generar el siguiente. Los datos son: código de responsable, motivo, código del producto, cantidad. 3.2. El actor ingresa el responsable. 3.3. El actor selecciona el motivo. 3.4. El actor selecciona el tipo de producto. 3.5. El actor ingresa cantidad del producto. 3.6. El actor selecciona "Buscar Producto" 3.7. El sistema muestra la interfaz Buscar Producto. 3.8. El actor selecciona un producto. 3.9. El sistema muestra regresa a la interfaz de Mantenimiento de Guía de Ingreso. 3.10. El actor graba. 4. El actor tiene la posibilidad de Modificar una Guía de Ingreso: 4.1. El sistema habilita la interfaz correspondiente para modificar una guía de ingreso. Los datos pueden ser: motivo, cantidad de productos, producto. 4.2. El actor modifica los datos presentados en la interfaz. 4.3. El actor graba los cambios. 4.4. El sistema presenta el listado de la guía de ingreso modificada. 5. Si el actor solicita Eliminar una Guía de Ingreso: 5.1. El actor tiene la posibilidad de eliminar o de anular: 5.1.1. Si selecciona eliminar: 5.1.1.1. El actor elimina la guía de ingreso. 5.1.1.2. El sistema presenta la primera guía de ingreso. 5.1.2. Si selecciona anular: 5.1.2.1. El actor anula la guía de ingreso. 6. El sistema presenta la guía de ingreso anulada.

Flujo Alternativo	No hay flujos alternativos para este caso de uso.

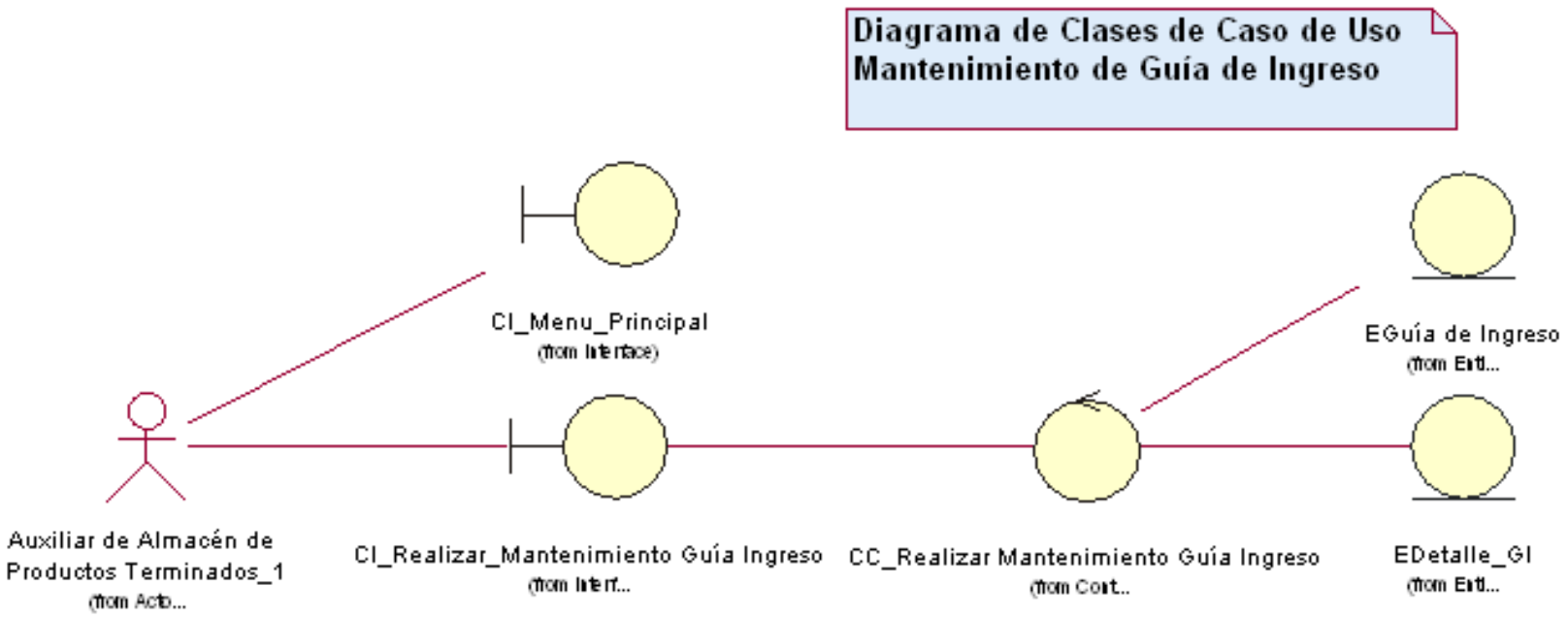
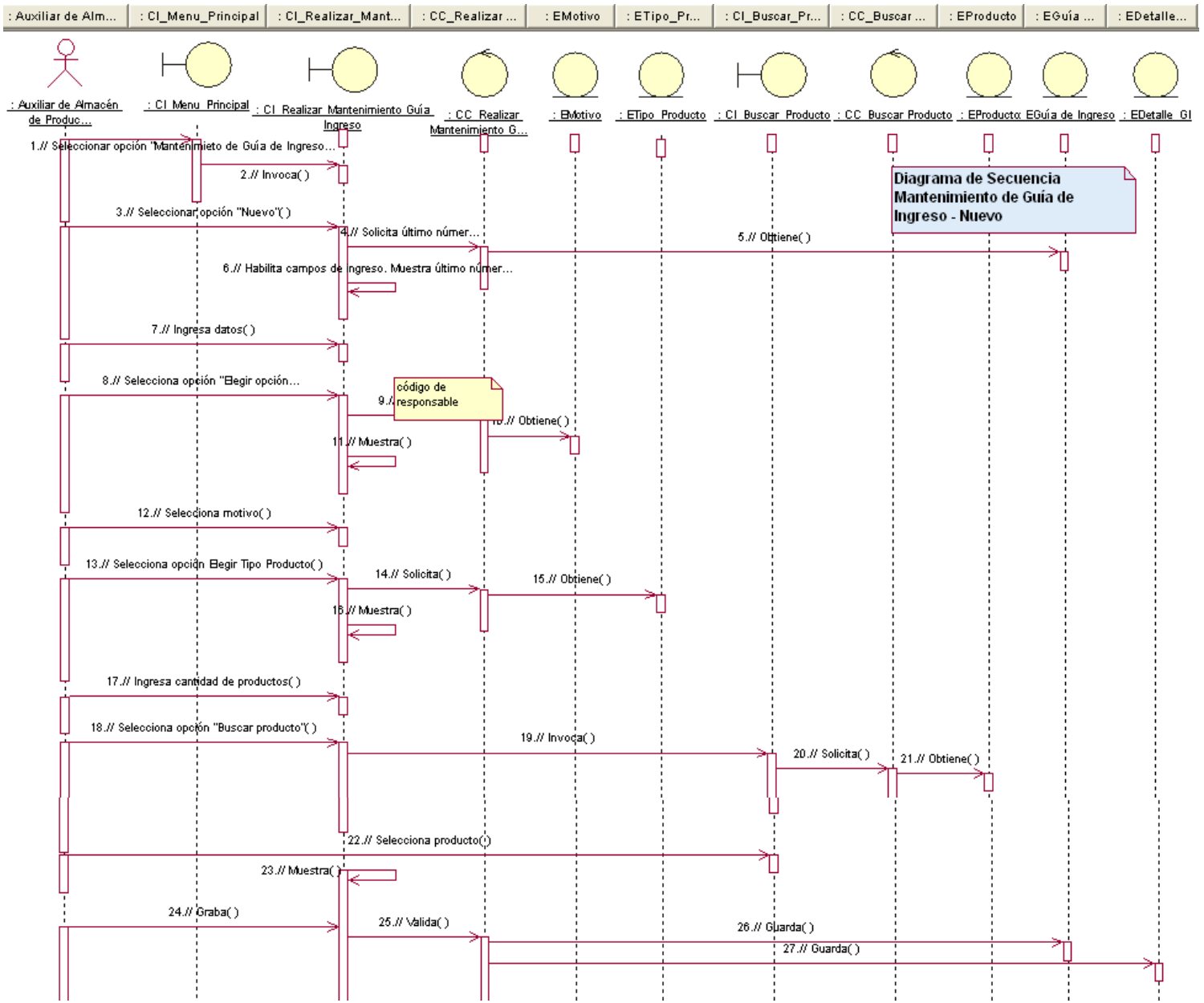


Diagrama de Clases de Caso de Uso
Mantenimiento de Guía de Ingreso
CI_Menu_Principal
(from Interface)
EGuía de Ingreso
(from Enti...
Auxiliar de Almacén de
Productos Terminados_1
(from Acto...
CI_Realizar_Mantenimiento Guía Ingreso
(from Interf...
CC_Realizar Mantenimiento Guía Ingreso
(from Cont...
EDetalle_GI
(from Enti...


: Auxiliar de Alm...
: CI_Menu_Principal
: CI_Realizar_Mant...
: CC_Realizar ...
: EMotivo
: ETipo_Pr...
: CI_Buscar_Pr...
: CC_Buscar ...
: EProducto
: EGuía ...
: EDetalle...
: Auxiliar de Almacén de Produc...
: CI_Menu_Principal
: CI_Realizar_Mantenimiento_Guía Ingreso
: CC_Realizar Mantenimiento G...
: EMotivo
: ETipo_Producto
: CI_Buscar_Producto
: CC_Buscar Producto
: EProducto
: EGuía de Ingreso
: EDetalle_GI
Diagrama de Secuencia Mantenimiento de Guía de Ingreso - Nuevo
1.// Seleccionar opción "Mantenimieto de Guía de Ingreso...
2.// Invoca( )
3.// Seleccionar opción "Nuevo"( )
4.// Solicita último númer...
5.// Obtiene( )
6.// Habilita campos de ingreso. Muestra último númer...
7.// Ingresa datos( )
8.// Selecciona opción "Elegir opción...
código de responsable
9.//
10.// Obtiene( )
11.// Muestra( )
12.// Selecciona motivo( )
13.// Selecciona opción Elegir Tipo Producto( )
14.// Solicita( )
15.// Obtiene( )
16.// Muestra( )
17.// Ingresa cantidad de productos( )
18.// Selecciona opción "Buscar producto"( )
19.// Invoca( )
20.// Solicita( )
21.// Obtiene( )
22.// Selecciona producto( )
23.// Muestra( )
24.// Graba( )
25.// Valida( )
26.// Guarda( )
27.// Guarda( )

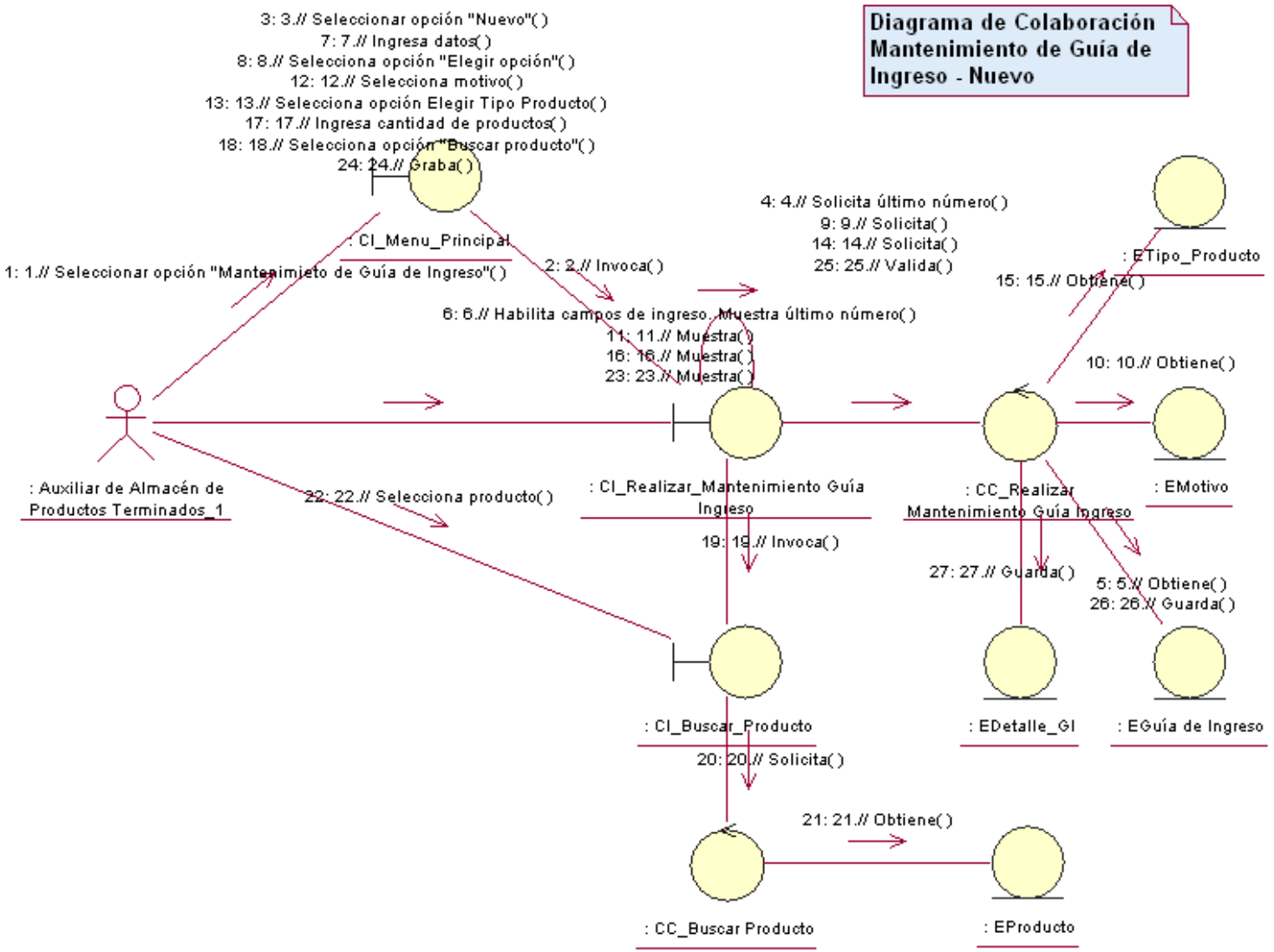
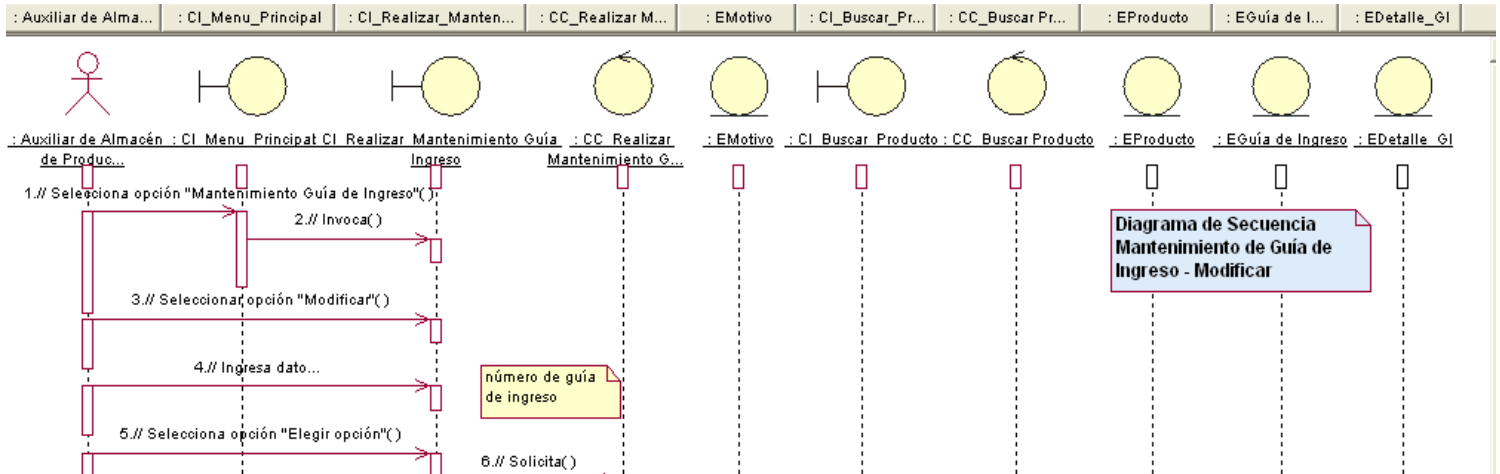


Diagrama de Colaboración Mantenimiento de Guía de Ingreso - Nuevo
3: 3.// Seleccionar opción "Nuevo"( )
7: 7.// Ingresa datos( )
8: 8.// Selecciona opción "Elegir opción"( )
12: 12.// Selecciona motivo( )
13: 13.// Selecciona opción Elegir Tipo Producto( )
17: 17.// Ingresa cantidad de productos( )
18: 18.// Selecciona opción "Buscar producto"( )
24: 24.// Graba( )
: CI_Menu_Principal
1: 1.// Seleccionar opción "Mantenimieto de Guía de Ingreso"( )
2: 2.// Invoca( )
4: 4.// Solicita último número( )
9: 9.// Solicita( )
14: 14.// Solicita( )
25: 25.// Valida( )
: ETipo_Producto
15: 15.// Obtiene( )
6: 6.// Habilita campos de ingreso, Muestra último número( )
11: 11.// Muestra( )
16: 16.// Muestra( )
23: 23.// Muestra( )
10: 10.// Obtiene( )
: Auxiliar de Almacén de Productos Terminados_1
22: 22.// Selecciona producto( )
: CI_Realizar_Mantenimiento Guía Ingreso
: CC_Realizar Mantenimiento Guía Ingreso
: EMotivo
19: 19.// Invoca( )
27: 27.// Guarda( )
5: 5.// Obtiene( )
26: 26.// Guarda( )
: CI_Buscar_Producto
: EDetalle_GI
: EGuía de Ingreso
20: 20.// Solicita( )
21: 21.// Obtiene( )
: CC_Buscar Producto
: EProducto


: Auxiliar de Alma...
: CI_Menu_Principal
: CI_Realizar_Manten...
: CC_Realizar M...
: EMotivo
: CI_Buscar_Pr...
: CC_Buscar Pr...
: EProducto
: EGuía de I...
: EDetalle_GI
: Auxiliar de Almacén de Produc...
: CI_Menu_Principal
CI_Realizar_Mantenimiento Guía Ingreso
: CC_Realizar Mantenimiento G...
: EMotivo
: CI_Buscar_Producto
: CC_Buscar Producto
: EProducto
: EGuía de Ingreso
: EDetalle_GI
1.// Selecciona opción "Mantenimiento Guía de Ingreso"( )
2.// Invoca( )
3.// Seleccionar opción "Modificar"( )
4.// Ingresa dato...
número de guía de ingreso
5.// Selecciona opción "Elegir opción"( )
6.// Solicita( )
Diagrama de Secuencia Mantenimiento de Guía de Ingreso - Modificar

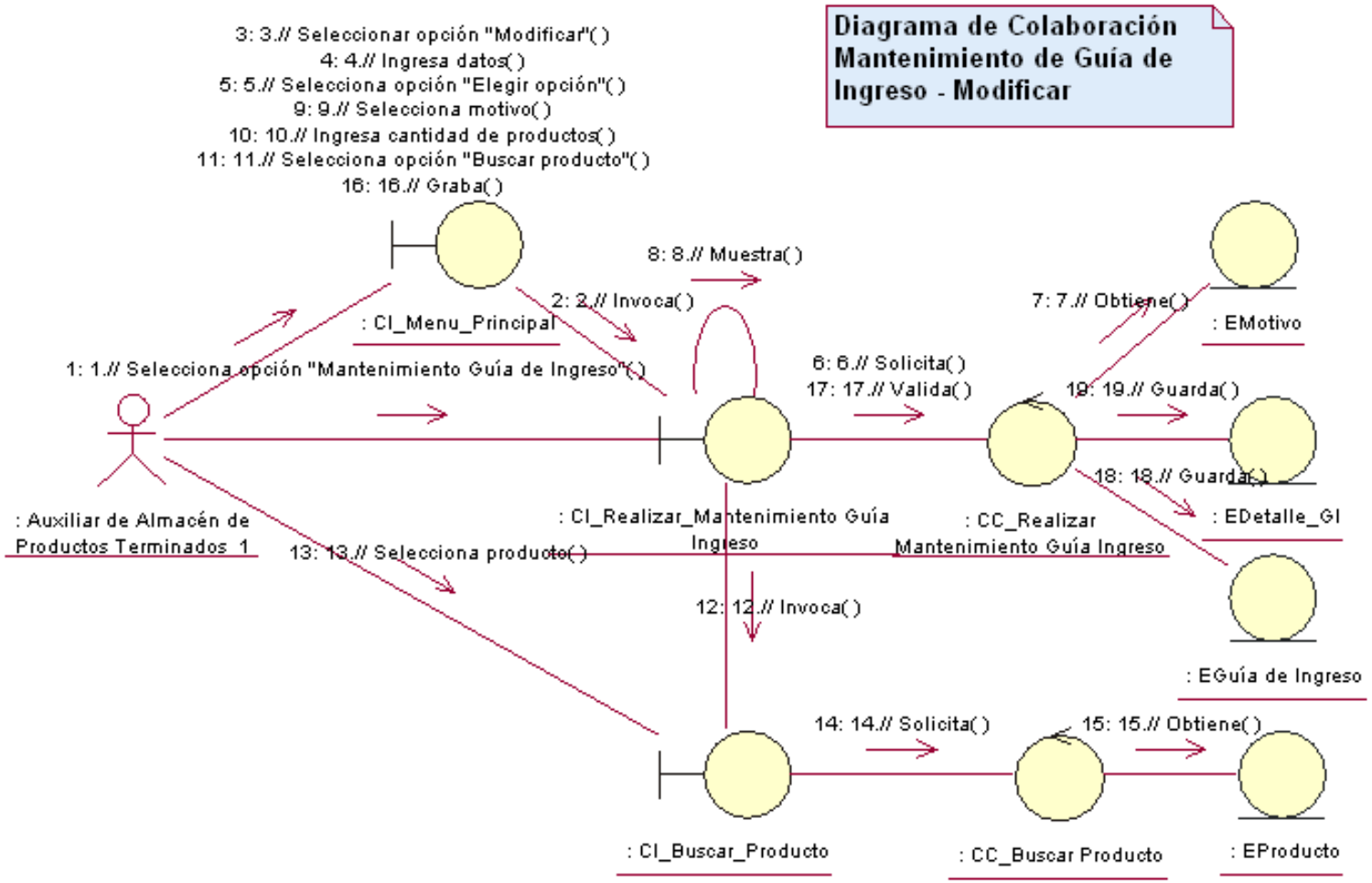
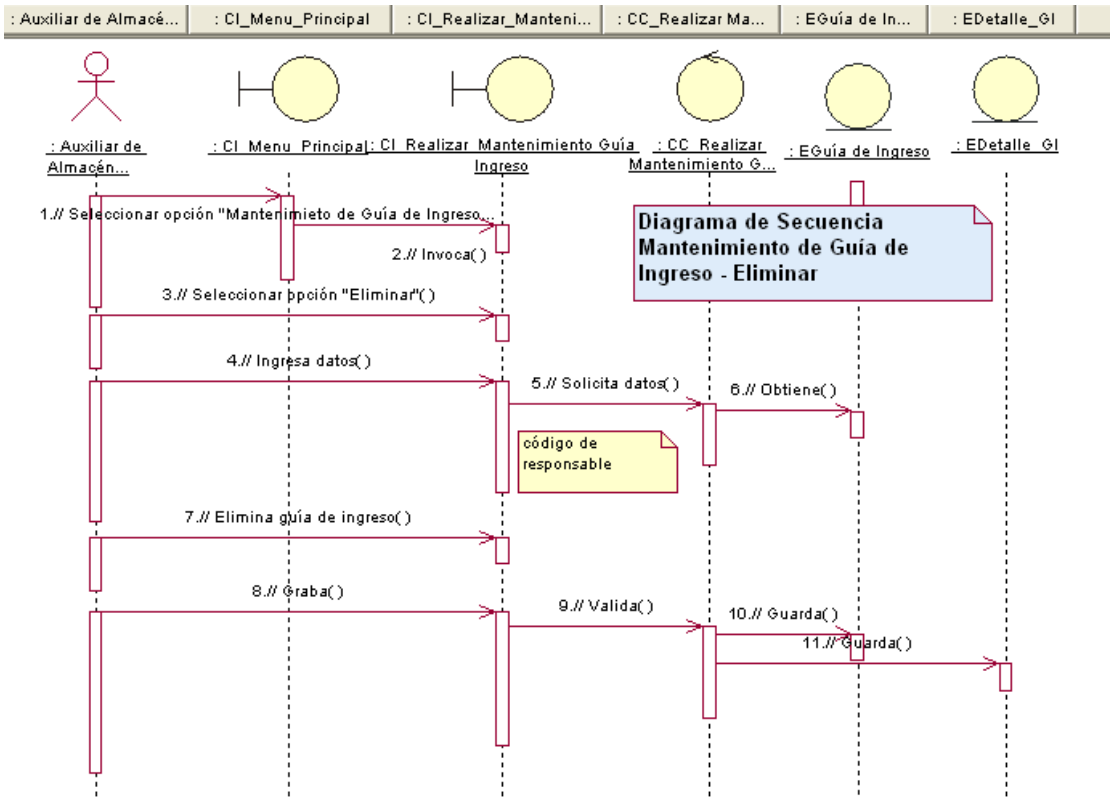
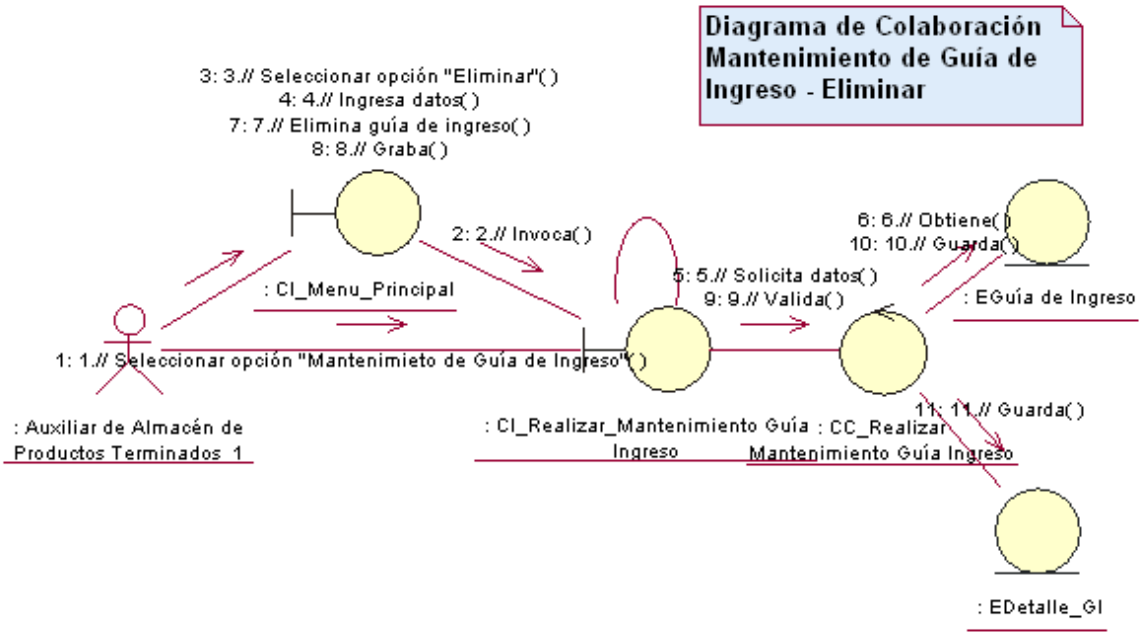


Diagrama de Colaboración
Mantenimiento de Guía de Ingreso - Modificar
3: 3.// Seleccionar opción "Modificar"( )
4: 4.// Ingresa datos( )
5: 5.// Selecciona opción "Elegir opción"( )
9: 9.// Selecciona motivo( )
10: 10.// Ingresa cantidad de productos( )
11: 11.// Selecciona opción "Buscar producto"( )
16: 16.// Graba( )
8: 8.// Muestra( )
2: 2.// Invoca( )
7: 7.// Obtiene( )
: CI_Menu_Principal
: EMotivo
1: 1.// Selecciona opción "Mantenimiento Guía de Ingreso"( )
6: 6.// Solicita( )
17: 17.// Valida( )
19: 19.// Guarda( )
18: 18.// Guarda( )
: Auxiliar de Almacén de Productos Terminados 1
: CI_Realizar_Mantenimiento Guía Ingreso
: CC_Realizar Mantenimiento Guía Ingreso
: EDetalle_GI
13: 13.// Selecciona producto( )
12: 12.// Invoca( )
: EGuía de Ingreso
14: 14.// Solicita( )
15: 15.// Obtiene( )
: CI_Buscar_Producto
: CC_Buscar Producto
: EProducto





Descripción de los procesos de sistema

f) Especificación del Negocio – Realizar Mantenimiento de Guía de Salida

Descripción	Subproceso de **Mantenimiento de Guía de Salida**.
Metas	Registrar Guía de Salida. Actualizar Guía de Salida. Eliminar Guía de Salida.

Flujo de Trabajo	1. Este caso de uso se inicia con la selección de la opción "Mantenimiento de la Guía de Salida" del Menú Principal. 2. El sistema muestra la interfaz de Guía de Salida. 3. Si el actor solicita registrar un Nuevo Guía de Salida: 3.1. El sistema habilita la interfaz correspondiente para registrar una guía de ingreso y solicita el último número para generar el siguiente. Los datos son: código de responsable, motivo, código del producto, cantidad. 3.2. El actor ingresa el responsable. 3.3. El actor selecciona el motivo. 3.4. El actor selecciona el tipo de producto. 3.5. El actor ingresa cantidad del producto. 3.6. El actor selecciona "Buscar Producto" 3.7. El sistema muestra la interfaz Buscar Producto. 3.8. El actor selecciona un producto. 3.9. El sistema muestra regresa a la interfaz de Mantenimiento de Guía de Salida. 3.10. El actor graba. 4. El actor tiene la posibilidad de Modificar una Guía de Salida: 4.1. El sistema habilita la interfaz correspondiente para modificar una guía de ingreso. Los datos pueden ser: motivo, cantidad de productos, producto. 4.2. El actor modifica los datos presentados en la interfaz. 4.3. El actor graba los cambios. 4.4. El sistema presenta el listado de la guía de ingreso modificada. 5. Si el actor solicita Eliminar una Guía de Salida: 5.1. El actor tiene la posibilidad de eliminar o de anular: 5.1.1. Si selecciona eliminar: 5.1.1.1. El actor elimina la guía de Salida. 5.1.1.2. El sistema presenta la primera guía de Salida. 5.1.2. Si selecciona anular: 5.1.2.1. El actor anula la guía de Salida. 6. El sistema presenta la guía de Salida anulada.

Flujo Alternativo	No hay flujos alternativos para este caso de uso.

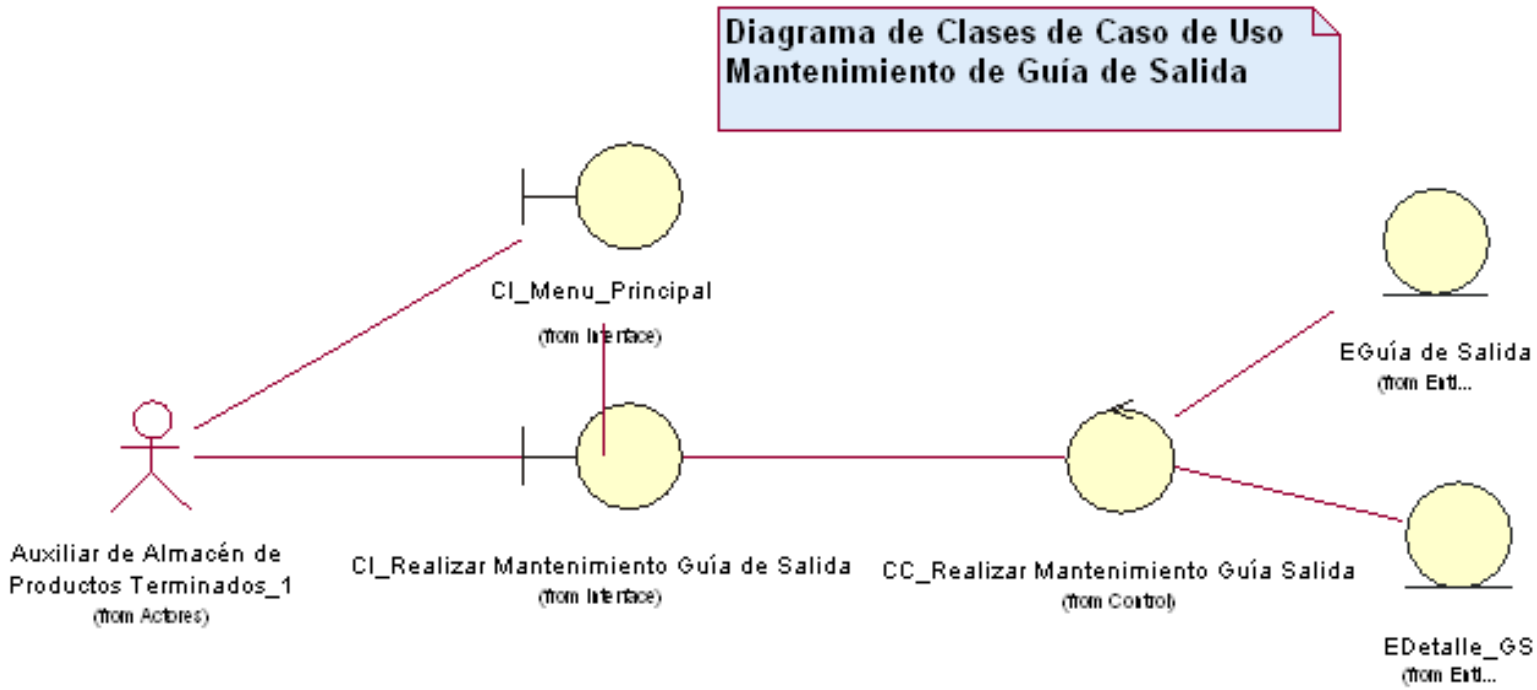
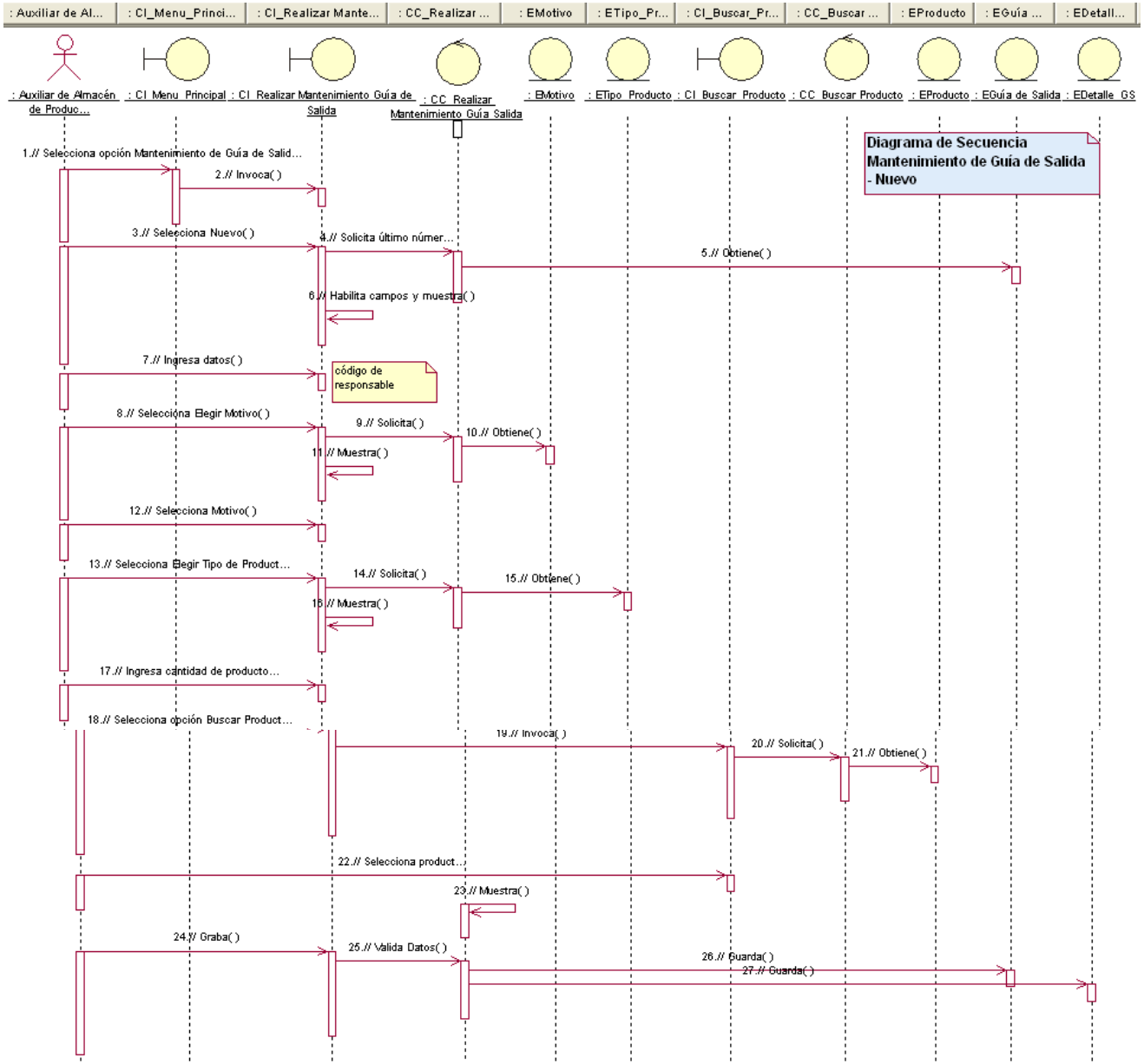
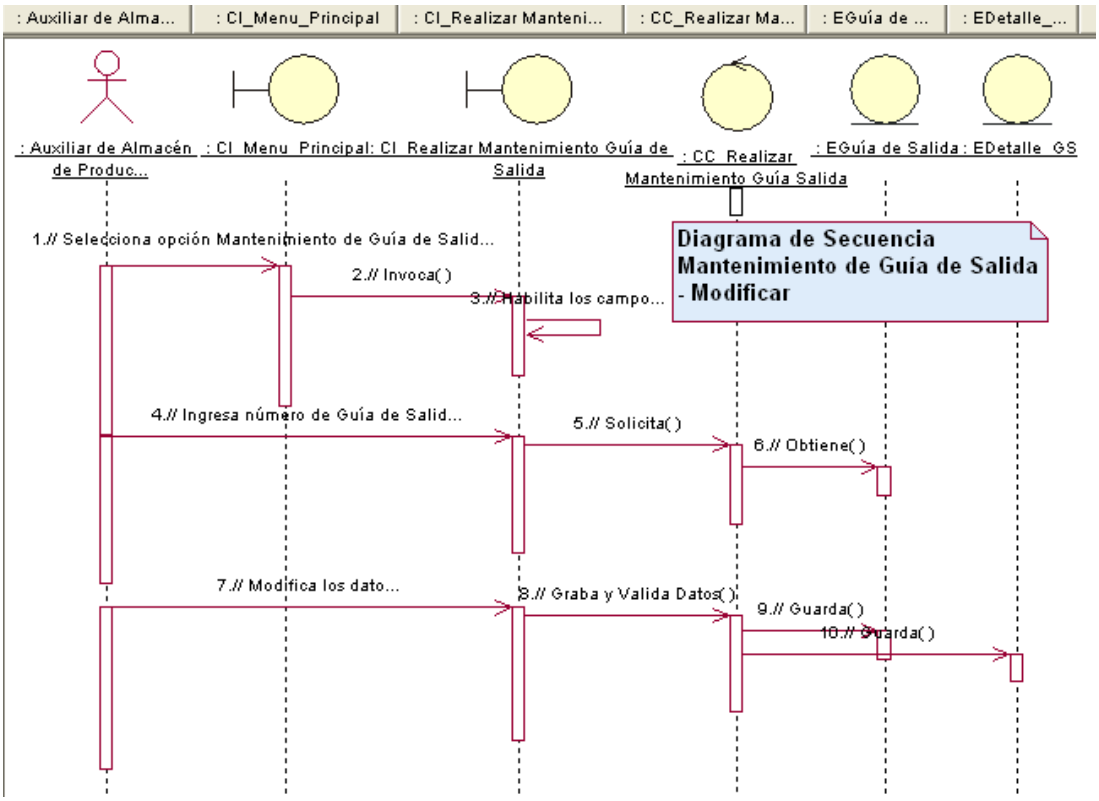
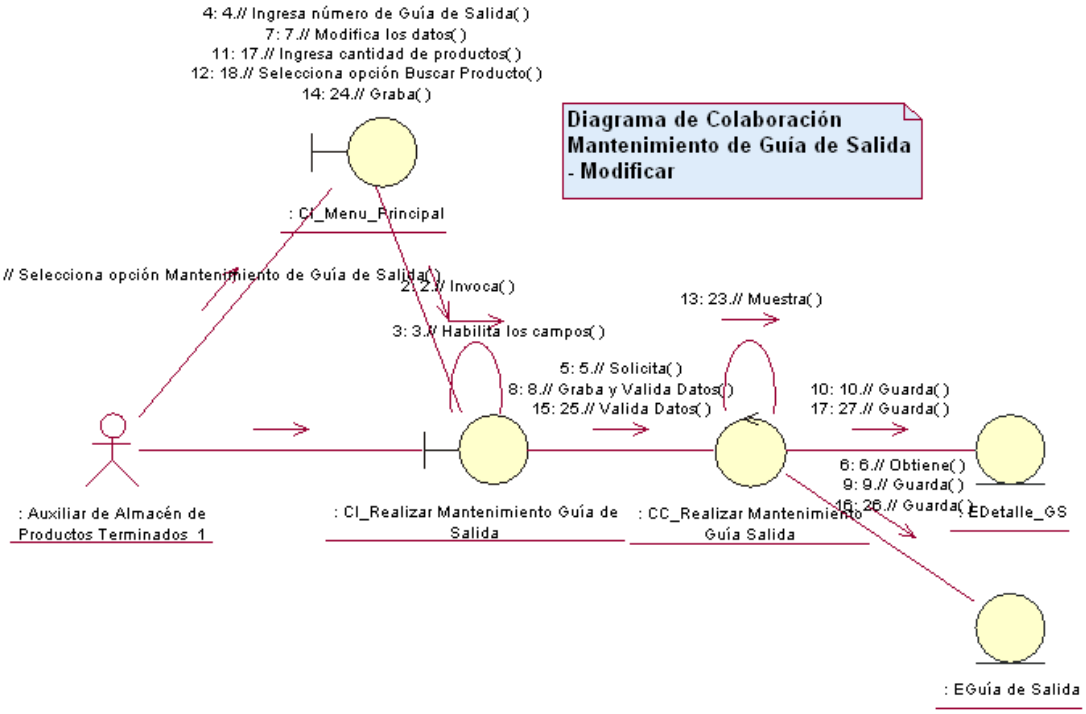
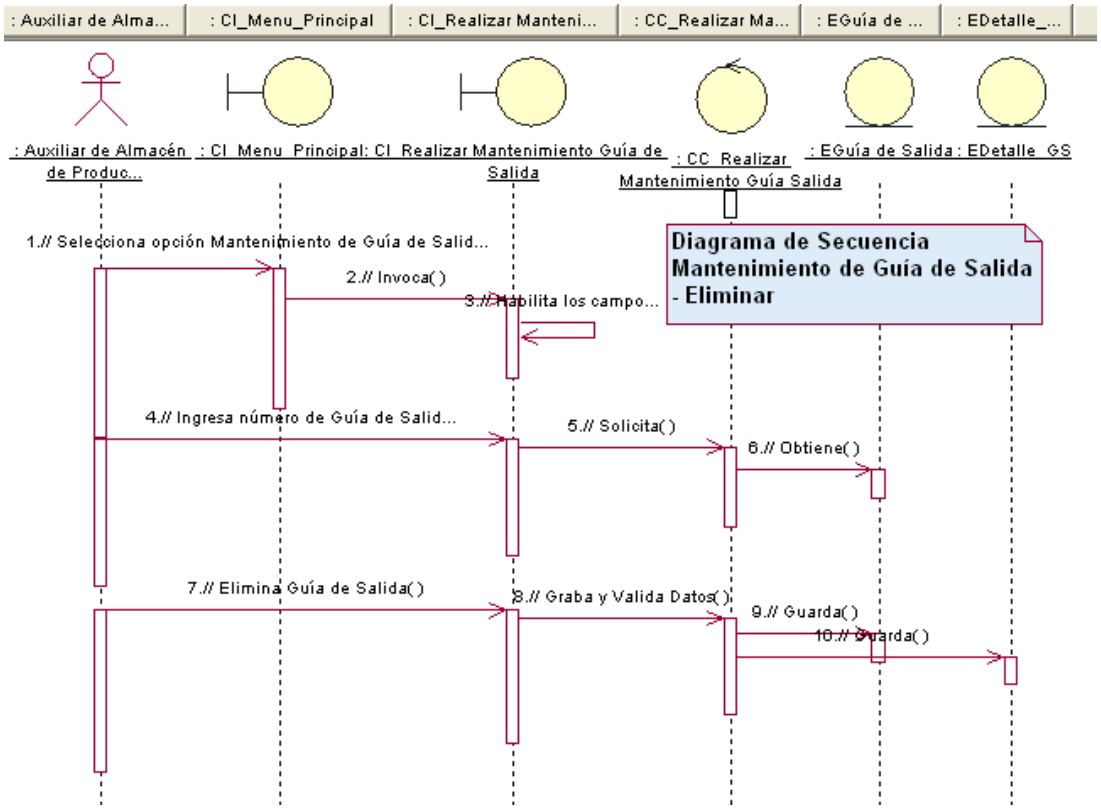


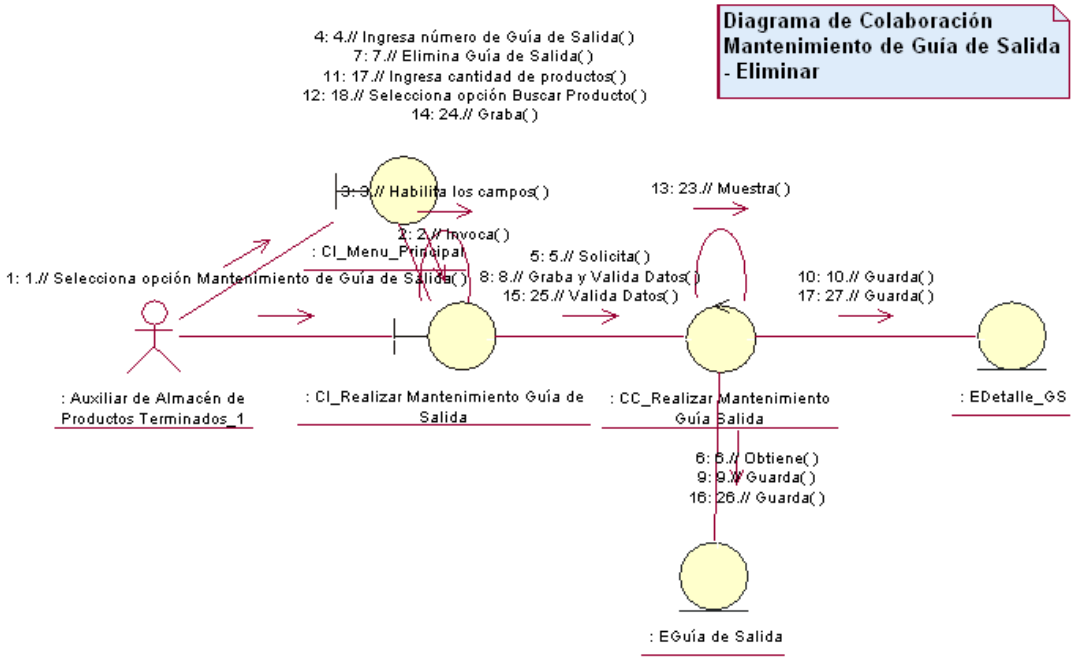
Diagrama de Clases de Caso de Uso
Mantenimiento de Guía de Salida
CI_Menu_Principal
(from Interface)
EGuía de Salida
(from Enti...
Auxiliar de Almacén de
Productos Terminados_1
(from Actores)
CI_Realizar Mantenimiento Guía de Salida
(from Interface)
CC_Realizar Mantenimiento Guía Salida
(from Control)
EDetalle_GS
(from Enti...


: Auxiliar de Al...
: Cl_Menu_Princi...
: Cl_Realizar Mante...
: CC_Realizar ...
: EMotivo
: ETipo_Pr...
: Cl_Buscar_Pr...
: CC_Buscar ...
: EProducto
: EGuía ...
: EDetall...
: Auxiliar de Almacén de Produc...
: Cl_Menu_Principal
: Cl_Realizar Mantenimiento Guía de Salida
: CC_Realizar Mantenimiento Guía Salida
: EMotivo
: ETipo_Producto
: Cl_Buscar_Producto
: CC_Buscar Producto
: EProducto
: EGuía de Salida
: EDetalle_GS
Diagrama de Secuencia
Mantenimiento de Guía de Salida
- Nuevo
1.// Selecciona opción Mantenimiento de Guía de Salid...
2.// Invoca( )
3.// Selecciona Nuevo( )
4.// Solicita último númer...
5.// Obtiene( )
6.// Habilita campos y muestra( )
7.// Ingresa datos( )
código de responsable
8.// Selecciona Elegir Motivo( )
9.// Solicita( )
10.// Obtiene( )
11.// Muestra( )
12.// Selecciona Motivo( )
13.// Selecciona Elegir Tipo de Product...
14.// Solicita( )
15.// Obtiene( )
16.// Muestra( )
17.// Ingresa cantidad de producto...
18.// Selecciona opción Buscar Product...
19.// Invoca( )
20.// Solicita( )
21.// Obtiene( )
22.// Selecciona product...
23.// Muestra( )
24.// Graba( )
25.// Valida Datos( )
26.// Guarda( )
27.// Guarda( )


: Auxiliar de Alma...
: CI_Menu_Principal
: CI_Realizar Manteni...
: CC_Realizar Ma...
: EGuía de ...
: EDetalle_...
: Auxiliar de Almacén de Produc...
: CI_Menu_Principal
: CI_Realizar Mantenimiento Guía de Salida
: CC_Realizar Mantenimiento Guía Salida
: EGuía de Salida
: EDetalle_GS
Diagrama de Secuencia Mantenimiento de Guía de Salida - Modificar
1.// Selecciona opción Mantenimiento de Guía de Salid...
2.// Invoca( )
3.// Habilita los campo...
4.// Ingresa número de Guía de Salid...
5.// Solicita( )
6.// Obtiene( )
7.// Modifica los dato...
8.// Graba y Valida Datos( )
9.// Guarda( )
10.// Guarda( )


4: 4.// Ingresa número de Guía de Salida( )
7: 7.// Modifica los datos( )
11: 17.// Ingresa cantidad de productos( )
12: 18.// Selecciona opción Buscar Producto( )
14: 24.// Graba( )
Diagrama de Colaboración Mantenimiento de Guía de Salida - Modificar
: CI_Menu_Principal
// Selecciona opción Mantenimiento de Guía de Salida( )
2: 2.// Invoca( )
3: 3.// Habilita los campos( )
13: 23.// Muestra( )
5: 5.// Solicita( )
8: 8.// Graba y Valida Datos( )
15: 25.// Valida Datos( )
10: 10.// Guarda( )
17: 27.// Guarda( )
6: 6.// Obtiene( )
9: 9.// Guarda( )
16: 26.// Guarda( )
: Auxiliar de Almacén de Productos Terminados 1
: CI_Realizar Mantenimiento Guía de Salida
: CC_Realizar Mantenimiento Guía Salida
: EDetalle_GS
: EGuía de Salida


: Auxiliar de Alma...
: CI_Menu_Principal
: CI_Realizar Manteni...
: CC_Realizar Ma...
: EGuía de ...
: EDetalle_...
: Auxiliar de Almacén de Produc...
: CI_Menu_Principal: CI_Realizar Mantenimiento Guía de Salida
: CC_Realizar Mantenimiento Guía Salida
: EGuía de Salida
: EDetalle_GS
1.// Selecciona opción Mantenimiento de Guía de Salid...
2.// Invoca( )
3.// Habilita los campo...
4.// Ingresa número de Guía de Salid...
5.// Solicita( )
6.// Obtiene( )
7.// Elimina Guía de Salida( )
8.// Graba y Valida Datos( )
9.// Guarda( )
10.// Guarda( )
Diagrama de Secuencia
Mantenimiento de Guía de Salida
- Eliminar


4: 4.// Ingresa número de Guía de Salida( )
7: 7.// Elimina Guía de Salida( )
11: 17.// Ingresa cantidad de productos( )
12: 18.// Selecciona opción Buscar Producto( )
14: 24.// Graba( )
Diagrama de Colaboración
Mantenimiento de Guía de Salida
- Eliminar
3: 3.// Habilita los campos( )
13: 23.// Muestra( )
2: 2.// Invoca( )
: CI_Menu_Principal
5: 5.// Solicita( )
1: 1.// Selecciona opción Mantenimiento de Guía de Salida( )
8: 8.// Graba y Valida Datos( )
15: 25.// Valida Datos( )
10: 10.// Guarda( )
17: 27.// Guarda( )
: Auxiliar de Almacén de Productos Terminados_1
: CI_Realizar Mantenimiento Guía de Salida
: CC_Realizar Mantenimiento Guía Salida
: EDetalle_GS
6: 6.// Obtiene( )
9: 9.// Guarda( )
16: 26.// Guarda( )
: EGuía de Salida

B. ESPECIFICACIONES DE REQUERIMIENTOS DE SOFTWARE

Requerimientos Funcionales

- Sistemas Operativos: Plataforma Windows.
- Base de datos en SQL Server 2000.
- La aplicación de Sistema Heredado será en Visual Basic 6.0.
- La aplicación de *Web Services* será en ASP con Visual Basic Script.
- El conector Wrapper será desarrollado en Visual Basic 6.0.

ANEXO 2

Interfaces de la Aplicación Web

Interfaces de la aplicación Web

A continuación se muestra la pantalla de mantenimiento de compras, en el cual se puede registrar modificar y eliminar registros de compra existentes.

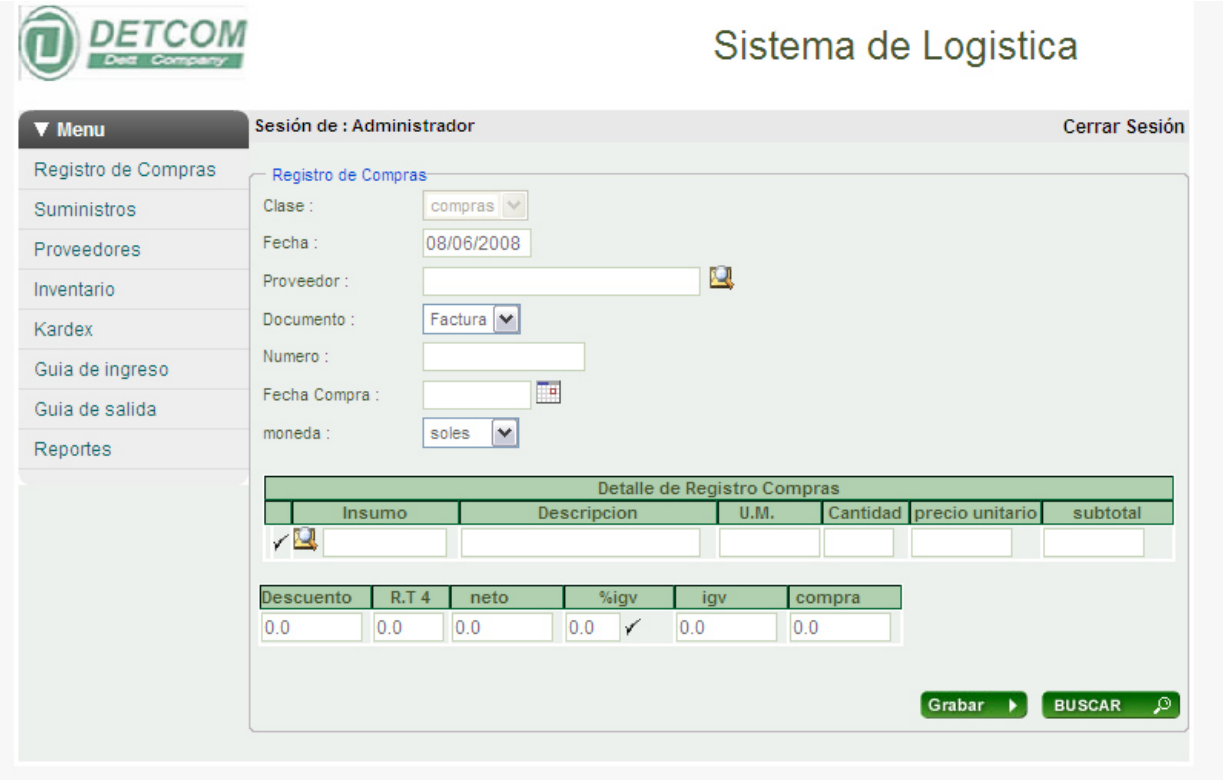


Figura A2.1: *Pantalla de mantenimiento de registros de oferta.*

Fuente: Creación Propia.

La siguiente opción de menú del sistema es la de mantenimiento de suministros, en la cual se deben ingresar detalles de los insumos como son las unidades de medida de los productos, precio y las unidades en stock.

Figura A2.2: *Pantalla de mantenimiento de suministros.*

Fuente: Creación Propia.

La siguiente opción de menú del sistema es la de mantenimiento de proveedores, la pantalla inicial muestra la consulta de los proveedores registrados en el sistema.

Figura A2.3: *Pantalla de consulta de proveedores.*

Fuente: Creación Propia.

Se puede registrar nuevos proveedores, para ello se deben ingresar detalles como son la razón social, la dirección, DNI, RUC, Email, teléfono, fax, etc.

Figura A2.4: *Pantalla de mantenimiento de proveedores.*

Fuente: Creación Propia.

La siguiente opción de menú del sistema es la de mantenimiento del inventario, en el cual se puede realizar consultas de los productos por almacén, así mismo se pueden realizar modificaciones o eliminaciones de artículos existentes.

Figura A2.5: *Pantalla de mantenimiento del inventario de productos.*

Fuente: Creación Propia.

La siguiente opción de menú del sistema es la de consulta del Kardex, que consiste en mostrar los detalles de las guías de productos.

Figura A2.6: *Pantalla de consulta del Kardex por artículos.*

Fuente: Creación Propia.

La siguiente opción de menú del sistema es la de mantenimiento de las guías de ingreso.

Figura A2.7: *Pantalla de mantenimiento de las guías de ingreso.*

Fuente: Creación Propia.

La siguiente opción de menú del sistema es la de mantenimiento de las guías de salida.

Figura A2.8: *Pantalla de mantenimiento de las guías de salida.*

Fuente: Creación Propia.

Referencias Bibliográficas

[AB 2004] José Angel Banares, Pedro J. Álvarez, Conceptos y Arquitectura de Servicios Web, Universidad de Zaragoza, http://iaaa.cps.unizar.es/docencia/SW.html#Origen, (23/04/2008).

[EV 2001] Jaime E. Villate, Introducción al XML, Universidad de Oporto, http://quark.fe.up.pt/cursoxml/curso.html, (03/05/2008)

[PA 2008] Paul Asman, Legacy Wrapping, Federal Reserve Bank of New York, http://jerry.cs.uiuc.edu/plop/plop2k/proceedings/Asman/Asman.pdf, (10/05/2008).

[HM 1996] Sneed Harry M., Encapsulating Legacy Software for Use in Client/Server Systems, Universidad de Vienna, (2008), 1-15.

[DWL 1989] Dietrich, W.C./Nackman, L.R./Gracer, Saving a legacy with objects, (1989), 100-112.

[HM 2008] Sneed Harry M., Wrapping Legacy Software for Reuse in a SOA, Universidad de Vienna, (2008), 1-15

[HM 1998] Sneed Harry M., Rudolf Majnar, A Case Study in Software Wrapping, International Conference on Software Maintenance, (1998), 80-86.

[IBM 2004] Martin keen, Amit Acharya, Susan Bishop, Alan Hopkins, Sven Milinski, Paul Verschueren, Implementing an SOA Using an Enterprise Service Bus, (2004), 33-67.

[MI 2008] Microsoft Corporation, Applying Microsoft Patterns to Solve EAI Problems, 0.9, (2008), 50-60.

[MT 1994] Mowbray, T. Zahari, R., The Essential CORBA, John Wiley & Sons, New York, (1994), 1-20.

[W3C 2004] W3C, Web Services Architecture, (2004), 1-9.

[AE 2007] Jaime Cristian Acevedo Emaldia, Criterios de Selección para las Herramientas de Orquestación de Servicios Web, Universidad de Chile, (2007), Santiago de Chile-Chile.

[AM 2002] Jorge Abin de María, Integración de Aplicaciones encapsuladas para el desarrollo de Sistemas de Información Cooperativos, Universidad de la Republica, (2002), Montevideo-Uruguay.

[CS 2005] Nancy Noemí Cova Suazo, Orquestación de Servicios Web orientada a aspectos, Instituto Politécnico Nacional, (2005), México.

[OT 2006] Salvador Otón Tortosa, Propuesta de una Arquitectura de Software Basada en Servicios para la Implementación de Repositorios de Objetos de Aprendizaje Distribuidos, Universidad de Alcalá, (2006), Alcalá De Henares-España.